TEC-0003

AD-A259 882

US Army Corps of Engineers
Topographic Engineering Center

Terrain Modelling: Shortest Path, Drain Patterns, and Interspersed Contours

Christoph Witzgall
National Institute of Standards and Technology
Center for Computing and Advanced Mathematics
Gaithersburg, Maryland 20899

Randall S. Karalus
U.S. Army Corps of Engineers
Topographic Engineering Center
Fort Belvoir, Virginia 22060-5546

December 1991

TEC

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
December 1991

3. REPORT TYPE AND DATES COVERED
Technical 1 Oct 90 - 30 Sep 91

4. TITLE AND SUBTITLE
Terrain Modelling: Shortest Path, Drain Patterns, and Interspersed Contours

5. FUNDING NUMBERS
Accession # DA313898
PR 4A162784A855
TA B
WU 059
PE 62784

6. AUTHOR(S)
Christoph Witzgall, Ph.D.; & Randall S. Karalus

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Corps of Engineers
Topographic Engineering Center
Cude Building #2592 (CETEC-GL-TD)
Fort Belvoir, Virginia 22060-5546

8. PERFORMING ORGANIZATION REPORT NUMBER
TEC-0003

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Institute of Standards and Technology
Center for Computing and Advanced Mathematics
Gaithersburg, Maryland 20899

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Not subject to copyright.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A computational approach to the task of generating contour-true and drain-consistent cartographic terrain surfaces from given grid-digitized contour lines is proposed and demonstrated for a still somewhat limited class of such contours. This approach is as follows: Consider all grid points located between two adjacent contours, and for each such grid point determine a shortest path towards the lower contour with respect to a grid metric that penalizes proximity to either contour. The pattern of these shortest paths is interpreted as a drain (ridge) pattern or network. This pattern contains the major drain (ridge) lines as implied by the contour information and thus permits the extraction of such features directly from grid-digitized contours. In a further development, the pattern is utilized to construct a grid elevation matrix by prorating elevations along its paths. An alternate drain (ridge) pattern derives naturally from that elevation matrix. This suggests an iterative procedure, alternating between extracting a drain (ridge) pattern from a grid elevation matrix and then generating a new elevation matrix from such a pattern. This procedure will be described and demonstrated. It will be considered as a method for interspersing contour lines of intermediate elevations.

14. SUBJECT TERMS

automated cartography	blocked drain	digitized contour	drainage network
interspersed contour	elevation matrix	interpolation	drain pattern
surface generation	terrain modeling		

15. NUMBER OF PAGES
76

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

Table of Contents

List of Figures

Preface

The work reported here was supported by the U.S. Army Corps of Engineers Topographic Engineering Center (TEC) under Interagency Agreement MIPR E8790K151 with the National Institute of Standards and Technology (NIST). We thank the many friends and colleagues in both institutions who have offered encouragement, advice, and assistance, in particular, Alan Geralnick, who got us started on ARC/INFO, Paul G. Logan, R. Justin Simpson, all at TEC, Javier Bernal at NIST, and Douglas R. Shier at the College of William and Mary.

Terrain Modeling: Shortest Paths, Drain Patterns, and Interspersed Contours

Christoph Witzgall [1] and Randall S. Karalus [2]

1. Introduction

A major issue in automated cartography is the generation of a terrain surface from discrete irregularly placed elevation measurements and, occasionally, additional information such as ridge and drain lines, lake shores and river banks. Elevation input may derive from photogrammetric measurements or from digitizing the contours of a map sheet. Representation of the terrain surface is frequently desired in the form of an elevation matrix over a regular rectangular grid. In what follows, we consider the particular task of generating a terrain surface from digitized contour information, the product being a grid elevation matrix. Examples of such elevation matrices in standardized form are the Digital Terrain Elevation Data, or "DTED", of the Defense Mapping Agency (DMA) and the Digital Elevation Matrix, or "DEM", of the United States Geological Survey (USGS).

More precisely, we expect the contour lines to be given in

"grid-digitized"

form (see Figure 1), that is, as sequences of consecutively distinct grid neighbors in a prescribed regular rectangular grid. Grid points are

"grid neighbors"

if they are contained in a single unit cell of the grid. Thus they are either horizontal, vertical or diagonal neighbors.

An example for grid-digitized contour information can also be found in Figure 14, which represents a 3400 ft by 4400 ft area in the proximity of Mustang Mountain, Fort Huahuca, Arizona. Three contour lines run through this area at elevations of 4500 ft, 4525 ft, and 4550 ft, respectively, from bottom to top of the picture. They are defined on a 20 ft by 20 ft grid. Based on these three contours, we will demonstrate a new approach to finding drain lines, generating elevations, and interspersing contours.

To avoid possible confusion about the subject of this report, we mention the separate but related issue of surface representation as opposed to surface generation. Indeed, representing

[1] National Institute of Standards and Technology, Gaithersburg, MD 20899
[2] U.S. Army Corps of Engineers Topographic Engineering Center, Fort Belvoir, VA 22060-5546

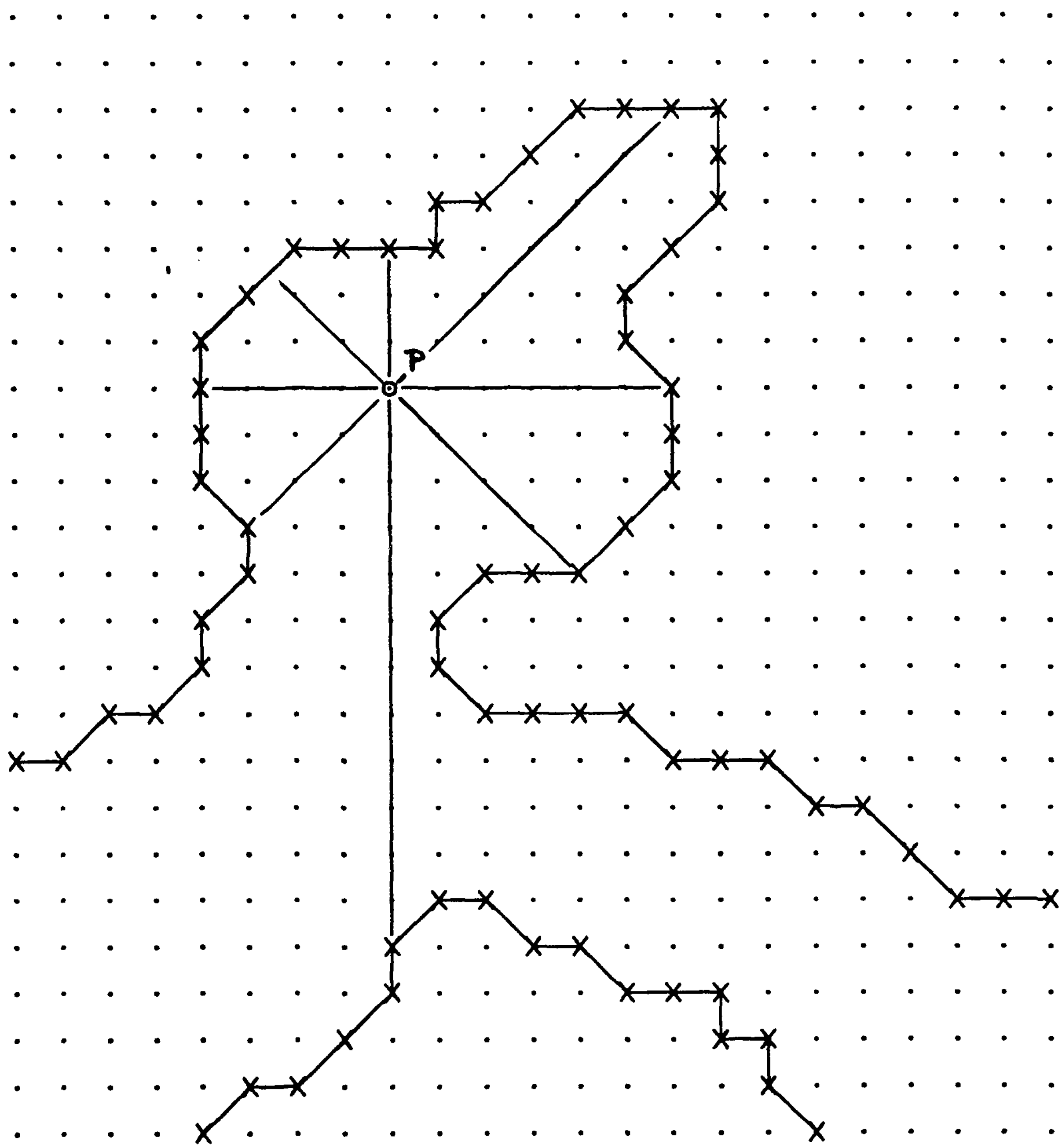

Figure 1: Square grid with two grid-digitized contour lines. The top contour is assumed to be at a higher elevation than the bottom contour. Elevation at a selected grid point P is often estimated from known elevations at bracketing post points, contour grid points that are closest horizontally, vertically, and diagonally to P.

a surface – once it is generated – by an unabridged elevation matrix may lead to a data set too voluminous for further processing such as visualization. One would want, therefore, to represent a surface by a more compact data structure ("data compression"). The work by Scarlatos (1990, 1991), for instance, addresses surface representation. In this report, however, we deal only with the issue of generating rather than representing surfaces.

1.1 Current methods. The first class of surface generation methods to be discussed here might be called "search-for-post" methods. For an arbitrary grid point P in the map area, these methods endeavor to estimate the elevation at that location from "post points" in its neighborhood, that is, grid points of given elevations such as grid points on grid-digitized contour lines. A typical approach is indicated in Figure 1. Here closest contour points bracketing grid point P along vertical, horizontal, and diagonal lines are identified. The elevation at location P is then estimated as a suitably weighted average of the known elevations at those post points (see for instance Rinehart and Coleman 1988). This basic approach may be enhanced by identifying more than two contour points in each search direction and then locally fitting a quadratic surface from which to gauge the desired elevation (Grotzinger, Danielson, Caldwell, and Mandel 1984).

An elegant and efficient recursive variation upon the above theme is sometimes referred to under the acronym PIPS (Noma 1974). This approach is also based on a regular rectangular grid of the map area, and requires contour lines in grid-digitized form. After establishing desired elevations along the boundary "neat lines" by interpolation, subsequent elevations are estimated along successive columns starting next to the bottom of, say, the second column of desired estimates. Each such elevation is estimated from neighbor elevations already set below and in a previous column, plus a suitably defined "closest" post point.

Increasingly popular are the so-called "triangulated irregular network" (TIN) methods for surface generation (see for instance Peucker(Poiker) and Douglas 1975). Typically, elevations are specified on an irregular finite set of "sites", including the corners of the map area. These sites are then "triangulated" by specifying a set of triangles such that: (i) for each triangle, its vertices are the only sites within the triangle or on its boundary; (ii) the triangles do not overlap; (iii) the triangles cover the map area. (This concept of triangulation bears, of course, no relationship to the geodetic activity of the same name.)

Any such set of sites may be triangulated in many different ways. "Delaunay" triangulation (see for instance Preparata and Shamos 1988) is most commonly used because of the ease with which it can be determined, because it is essentially unique, and because it tends to avoid long skinny triangles. Delaunay triangulations (Figure 2) are characterized by the fact that the circle circumscribed around any of the triangles contains no sites in its interior. However, there are other triangulation principles each of which, depending on the case at

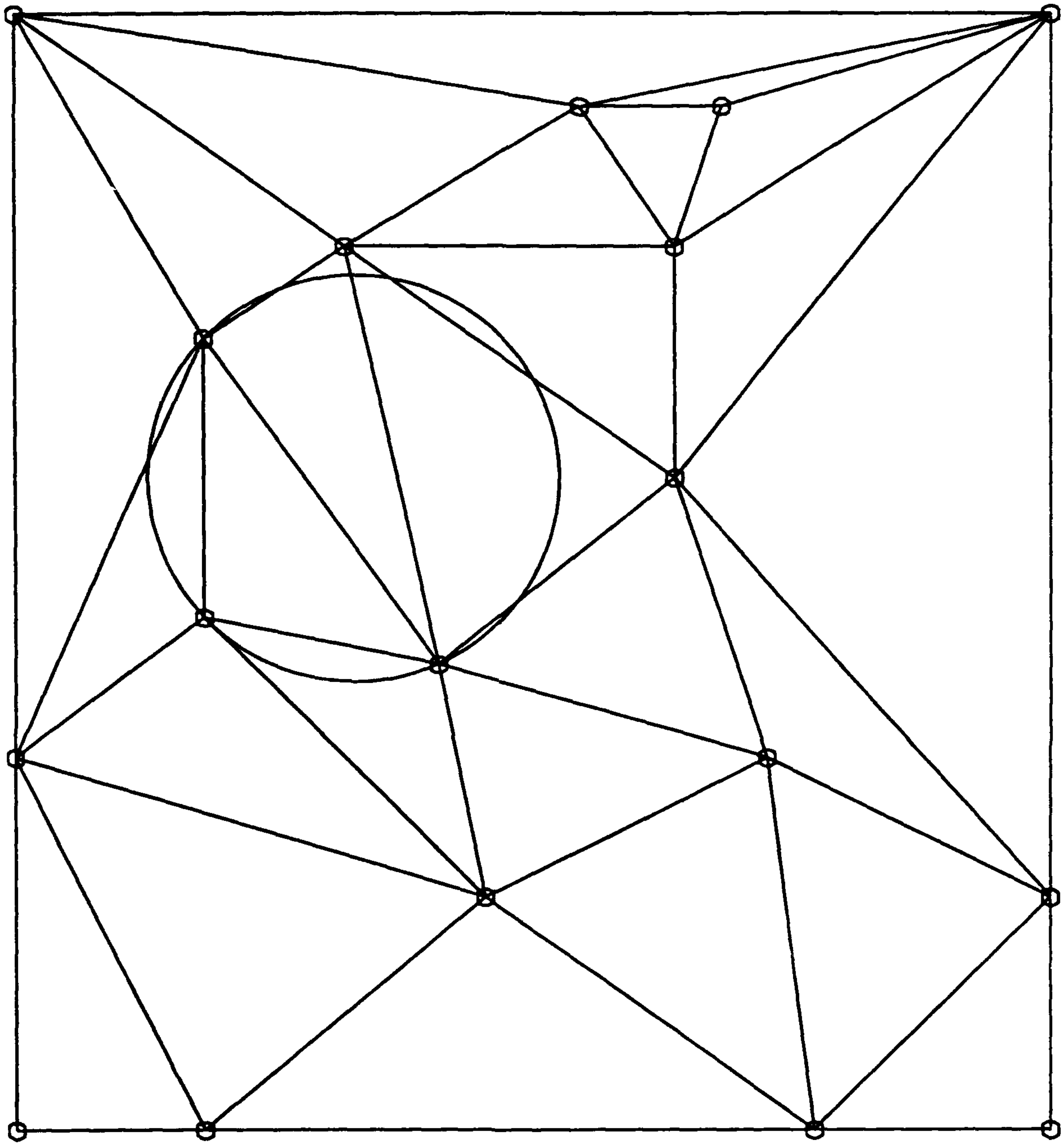

Figure 2: Set of sites, indicated by small circles, and their Delaunay triangulation. Circles circumscribed around triangles do not contain sites in their interior as evidenced by the circumscribed circle shown here.

hand, may be equally suitable or superior to the Delaunay principle as the basis for a TIN method.

Once a triangulation has been chosen, the typical TIN method passes planes through the elevated points at the vertices of each triangle, respectively, thus creating a continuous faceted surface which at each site meets the given site-specific elevation. This procedure is an instance of the well-known finite element (see for instance Zienkiewicz 1971) method for engineering applications. Here the "element" is a planar triangle in space. In order to find the elevation of the so constructed faceted surface at an arbitrary location in the map area, (i) the base triangle which contains that location must be found and (ii) the elevation at that location of the corresponding element must be determined. Generating a grid elevation matrix from the above surface specification proceeds essentially along the same lines.

The use of curved, higher order elements such as the Hsieh-Clough-Tocher element (see Clough and Tocher 1965, Lawson 1977) has been explored by Mandel, Witzgall and Bernal (1987). In that approach, the surface elements fit together smoothly, that is, without creases at their boundaries. The strength of this approach lies in its ability to include information about the general trend of the terrain into the local definition of the surface. In addition, it lends itself naturally to utilizing information about tangents to contour lines. Surfaces constructed in this fashion also depend less than surfaces built with planar elements on the particular way in which the given elevation sites are triangulated.

Many issues have yet to be explored before the respective merits of the above methods can be fully understood. We like the prospects of higher order TIN methods because we believe that future applications will require higher resolutions and more detail than is usually specified at present, say, for DTED generation, but it is still unclear to us what levels of accuracy will be needed.

There is also the question of whether proper test beds exist to validate such levels of accuracy. Errors may not only be due to the process of surface generation, but may also be inherent in the input data. The validity of the surface generation process should thus be measured not against the actual shape of the terrain, but rather against the shape of the terrain as specified by the input data.

1.2 Task definition. Since the level of accuracy desired for surface generation will surely vary for particular applications, and since conceptual uncertainties surround the issues of input validity and output validation, we seek a generic task definition based on the principle of

preservation of information.

By this we mean that the information and its implicit assumptions as presented by the cartographer should be fully recoverable. This principle can be verified, whereas "accuracy" may be elusive because of incomplete or erroneous initial information. In our case, explicit information consists of grid-digitized contour lines. Thus we will require that a surface be generated which is

"contour-true",

that is, a surface from which the course of the given contour lines can be recovered just by specifying their respective elevations. This requires not only that the generated surface assumes the specified elevations at contour locations, but also that elevations are higher on one side and lower on the other side in the immediate proximity of each given contour line.

Here we have assumed that the terrain, as viewed by the cartographer preparing the contour information, does not contain flat areas, that is, areas of constant elevations. If there are such areas, then contour lines at the corresponding elevations are not defined. What is defined are

"one-sided contour lines",

which bound a flat area either towards upper or lower terrain. The shore line of a small lake or the boundary of a plateau may have been included as contour lines in the overall contour information. Such contours are one-sided, and if they are not marked as such, the information whether their interior is to be treated as flat or curved is lost and cannot be retrieved by any terrain modeling method.

In addition to the actual contour information, there is implicit information about the behavior of the terrain between adjacent contour lines at different elevations. Suppose a rain drop is falling onto an arbitrary location on the upper contour. Then it is tacitly assumed that, by following steepest descent, the drop will be able to flow downhill until it reaches the lower contour. Moreover, it should be able to do so without major detours and without drastic changes in its rate of descent. In other words, the generated surface should be

"drain-consistent".

In particular, there should be no

"blocked drains",

that is, local minima surrounded by higher elevations on a major drainage line. Again we must presume at this point that, if the cartographer was aware of some exception to these tacit assumptions, he would have devised a way to let us know.

In order to make the notion of drain consistency more precise, we consider a "drain pattern" or "drainage network"

between contours in the grid (see Figure 3) of a given elevation matrix. Such a pattern results if every grid point between contours is assigned a

"successor grid point",

that is, a grid neighbor of steepest positive down-slope as indicated by the grid elevations in the given elevation matrix. This choice of grid neighbor indicates the direction of drainage from the grid point in question. Successively stepping from grid point to successor grid point determines a

"drain path"

in the grid. This grid path may be interpreted as the path of drainage from some given grid point.

The pattern formed by all the drain paths is a special kind of graph known as a "forest". All the paths that connect to the same contour point form a "tree". The grid points in each particular tree form its "nodes", the links which connect neighboring grid points in the tree form its "arcs". Fittingly, the forest is the totality of its trees. The point at which a tree meets the contour, is generally called its

"root".

We thus distinguish the

"root contour"

from the

"source contour".

Of the two adjacent contours considered here, the contour of higher elevation is the source contour, the one of lower elevation, the root contour.

Of particular interest in what follows are the

"dead ends"

of the forest. These are tree nodes each of which is joined by only one arc. In a sense they are the points which are furthest removed from their respective roots. In Figure 3, dead ends are marked by circles.

If some grid point, not located on the boundary of the grid or on one of the contours, lacks an assigned successor, then that grid point must represent the deepest point of a sink or else be mired in a flat spot. In the absence of such points, we call a drain pattern

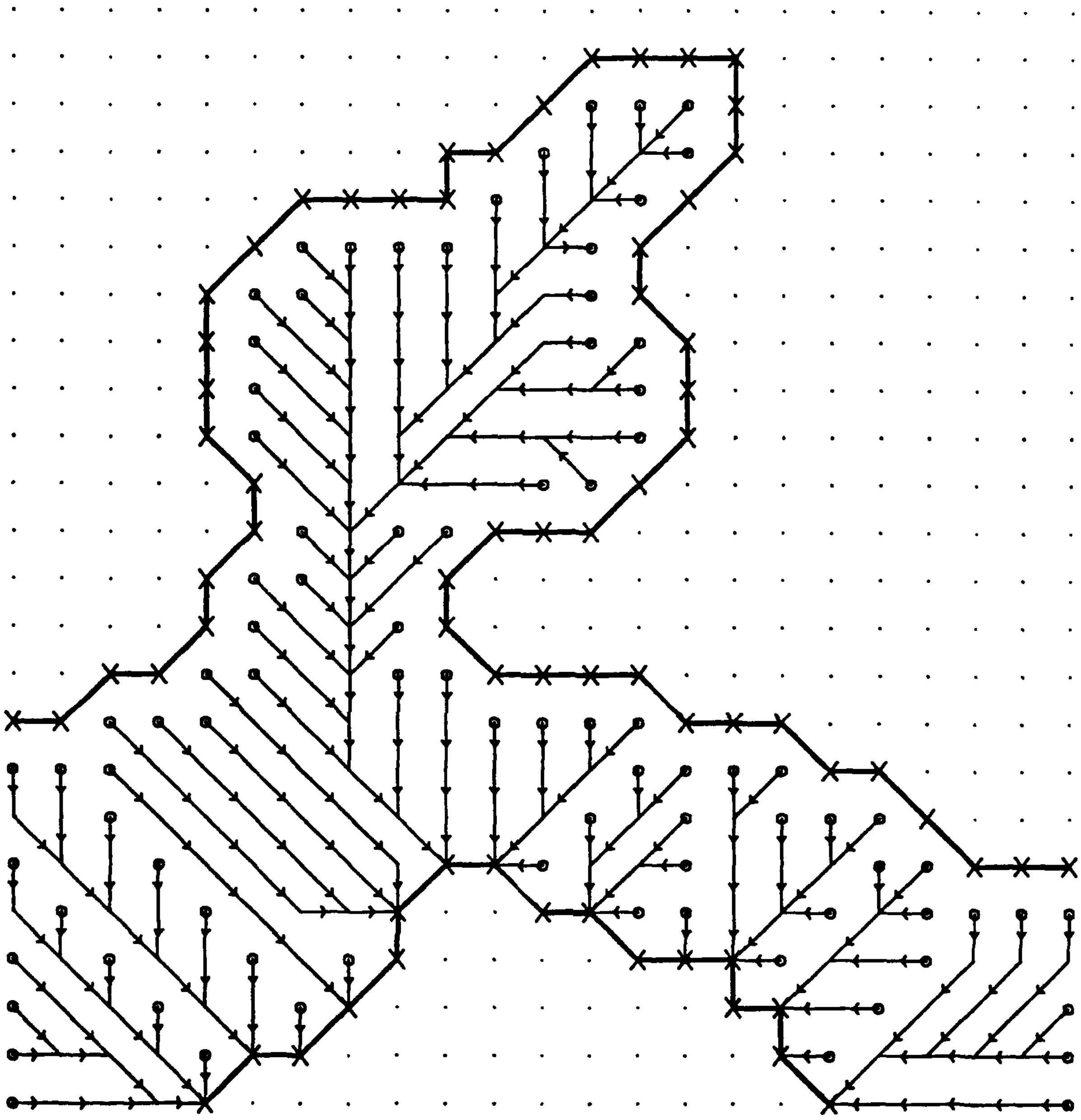

Figure 3: Inter-contour portion of a drain pattern. This drain pattern is complete in that it does not stop its descent before reaching the lower contour. In graph theoretical terms, this pattern represents a forest rooted at the lower contour. Dead ends are marked by circles. Due to reduction to grid points, the drain pattern is only approximately perpendicular to contours.

"complete".

If an inter-contour drain pattern is complete, then all of its trees have their roots on the root contour. The drain pattern shown in Figure 3 is complete in that sense.

For a generated surface or, more precisely, a grid elevation matrix to be drain consistent, it must give rise to a complete drain pattern.

We can now define the task at hand:

Given a regular rectangular grid of points covering the map area, and given in this grid a set of grid-digitized contours, we wish to construct a grid elevation matrix that is both contour-true and drain-consistent.

How do the previously mentioned methods measure up against that goal? In what follows, we will show that they do not meet the specifications unless special measures are taken.

Consider the classic TIN procedure: select a sample of contour points, construct a Delaunay triangulation with those sample points of known elevation as sites, and fit planar triangular patches to the vertex elevations of each triangle in the triangulation. Figure 4 illustrates the selection of sites from contour points and their subsequent triangulation. The upper contour in this example forms a bulge, and several triangles in the vicinity of that bulge take all three of their vertices from the same contour line. Those vertices are therefore of the same elevation. The corresponding surface patches are thus horizontally flat, creating a large flat spot or

"terrace".

As a result, the original shape of the contour line cannot be recovered from the generated surface, which is therefore not contour-true. Note that terrace formation could not have been avoided by selecting more triangulation sites on the contour line, in other words, by a denser sample. Terracing can be mitigated, but not entirely avoided by choosing a triangulation – in general no longer Delaunay – that takes into account the particular shape of the given contour lines. Blocked drains can in general be avoided if the triangulation is chosen such that contours do not intersect the interior of triangles. This can usually be assured by dense sampling. Even if blocked drains are absent, the elevation matrix may not be drain consistent according to our definition because the presence of terraces: the grid elevations do not characterize a complete drain pattern because some grid points will not have lower neighbors.

Terracing is also a problem for search-for-post methods, although to a lesser extent than for TIN methods. Blocked drains are generally avoided.

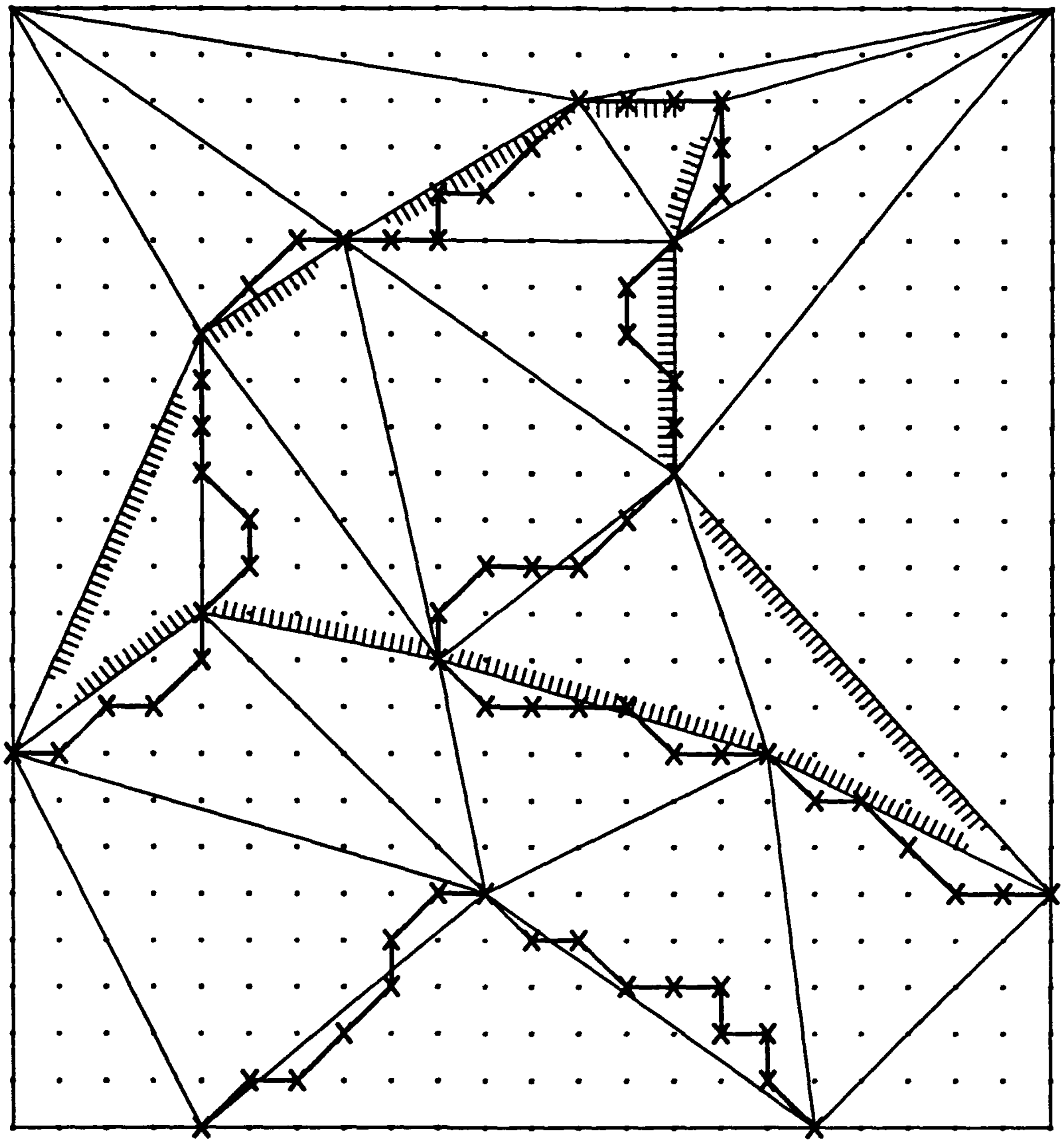

Figure 4: Triangulation of sites selected on contours and four grid corners. The vertices of triangles in the shaded area are sites from the same contour and thus are assigned the same elevation. A straight TIN method will therefore create a terrace in the shaded area.

If a TIN method with curved elements is used, terracing can be avoided if tangents to contours are utilized at the sample sites, and if the sample sites are sufficiently dense. But drain-consistency remains a problem. Indeed, higher order methods, because of their tendency to utilize terrain trends, are likely to create a sink in response to a widening followed by a narrowing of a valley as exhibited by the shape of the contour in Figure 1.

Some of the above difficulties can be resolved if feature information, such as drain lines, is available, regardless of the particular method used. However, to be able to intersperse additional contour lines before using, say, a TIN method would be particularly helpful. For this reason, we are emphasizing the potential of our approach for interspersing contour lines even more than its potential for generating elevation matrices. For instance, the shape and location of an interspersed contour line may well be more trustworthy than its associated elevation: by applying techniques inherent in the use of curved triangular elements as described in Mandel, Witzgall, and Bernal (1987), one can adjust the elevation of a contour line so as to best conform with the trend of the terrain without changing the shape and location of that line.

In a seminal paper, Christensen (1987) proposed an extension to the idea of a contour-oriented choice of the triangulation by utilizing the "medial axis" between two (complete) contour lines, that is, the locus of equal distance from contour points in the area between contour lines. After introducing elevations on the medial axis either at mid-elevation or suitably prorated, he constructs a triangulated surface that is both contour-true and drain-consistent. The medial axis concept is based on distances from contours, a concept that will be developed further and in a different direction in the work reported here.

1.3 Scope of the report. We propose a new approach to terrain modeling, based on shortest path techniques for the generation of complete drain patterns between contours in a rectangular grid. This approach permits the extraction of major drain lines from given contours alone, to be utilized within other more commonly used surface generation methods. We extend that approach to develop a surface generation method in its own right which meets the criteria of being contour-true and drain-consistent. We will demonstrate the approach by applying it to the problem of interspersing contour lines at intermediate elevations.

The problem of identifying drainage lines or, as they are also called, "channels" or "slope lines", has received much attention. The reader may want to consult Douglas (1987) and Seemuller (1989). The point of departure for these researchers has been the grid elevation matrix, whereas in our work drainage lines are derived from contour lines, prior to the determination of grid elevation matrices.

Our method proceeds in several stages. In what follows, each section in the main body of our report describes one of those stages. Details of the current implementation are given in the Appendix.

- Section 2: For each grid point, the shortest distance – stepping along successively neighboring points in grid – to a contour is determined.

- Section 3: Those distances are used to define new distances between neighboring points of the grid and thereby establish a new grid metric. In this new grid metric, distances are long in the proximity of contour points and short otherwise. For every grid point in the region between two adjacent contour lines, a shortest path to the root contour is determined. These shortest paths merge with one another upon approaching the lower boundary. They thus determine a complete drain pattern between the two contours and define major drain lines.

- Section 4: Given a complete drain pattern between two adjacent contour lines, elevations may be prorated along paths in the drain pattern leading from the source to root contour. Using such prorating schemes, elevations may be assigned to any grid point between the two contours. In other words, a grid elevation matrix may be constructed from drain patterns in areas bracketed by contours.

- Section 5: In turn, a grid elevation matrix gives rise to a drain pattern by connecting grid points to their neighbors of steepest descent. In general, this derived drain pattern does not duplicate the original drain pattern from which the elevation matrix was constructed. Indeed, while the paths in the drain pattern are downhill paths with respect to the elevation matrix, they are not necessarily the steepest downhill paths. The derived drain pattern can then be used to generate an additional elevation matrix, which gives rise to yet another drain pattern. This suggests an iterative procedure.

- Section 6: The extraction of intermediate contour lines from the elevation matrix generated in the previous step will be demonstrated for a 3400 ft by 4400 ft portion of a DMA data set describing the Mustang Mountain, Fort Huahuca, Arizona, area.

We hasten to warn the reader that this novel methodology is not yet mature and is far from ready for a production environment. At present, closed contours that neither enclose other contours nor contain spot elevations cannot be handled. Examples of such contours are the highest contour to surround an unmarked summit and the lowest contour in a depression. In fact, that problem extends to some cases of contours that are "almost" closed. Difficulties are also caused by contours that end in the interior of the map area rather than at the boundary. This happens, for instance, if contours on the source map are too close together because of steep terrain and therefore cannot be drawn individually. Thus further work is needed to extend the range of applicability of our method. Wider testing is required. Last but not least, this method is computationally expensive, although highly parallel.

Also, present software requires the user to specify ahead of time which contours are adjacent to each other. For large contour data sets, adjacency is often not recorded and cannot be ascertained by manual inspection. A solution to this pesky data processing problem, however, promises to be gained as a spin-off from the distance determination process described in Section 2.

2. Distances from contour lines

Given a rectangular grid of square unit cells, we define a

$$\text{“grid path” } P(\underline{k}, \bar{k}) \;=\; \{\underline{k} = k_0,\; k_1,\; \ldots,\; k_n = \bar{k}\}$$

between two grid points $\underline{k}$ and $\bar{k}$ as a sequence of grid points such that two consecutive grid points, k_{i-1} and k_i, are grid neighbors. The grid-digitized contour lines in Figure 1 are examples of grid paths. Since square unit cells have been assumed, we consider the

$$\text{“natural grid distance” } \operatorname{dist}(k_{i-1}, k_i)$$

between horizontal and vertical neighbors to be 1 and the distance between diagonal neighbors to be $\sqrt{2}$. This definition of distance between grid neighbors establishes a

“natural grid metric”

in which to measure lengths of grid paths and distances between any two grid points. Thus the length of a grid path is the sum of the length of the distances between consecutive points in the path $P(k \;\, ^{-})$:

$$\operatorname{length} P(\underline{k}, \bar{k}) \;=\; \sum_{i=1}^{n} \operatorname{dist}(k_{i-1}, k_i).$$

More generally, the

$$\text{“natural grid distance” } \operatorname{dist}(\underline{k}, \bar{k})$$

between two arbitrary grid points $\underline{k}$ and $\bar{k}$ is then defined as the length of a shortest path between those two points in the natural grid metric. In Section 3, we will introduce an alternate grid metric, from which we will derive a different length and distance measure.

Determining a shortest path between two grid points, and thereby the distance of these two points, is an instance of the problem of finding a shortest path in a network or "graph". This is a classical problem of graph theory, and many excellent algorithms have been proposed for its solution (see for instance Witzgall, Gilsinn, and Shier 1981).

Here, we are mainly interested in the fact that the shortest paths in our present pattern define for each grid point k its

"natural grid distance from contours" dist(k),

that is, the length of a shortest path – there may be several – from grid point k to a contour point that is closest in the natural grid metric.

The following discussion of how to find such shortest paths from contours is intended to illustrate some of the salient features common to almost all algorithms for determining shortest paths; it does not describe the method actually implemented (See Appendix).

For any grid point in the immediate neighborhood of a contour grid point, a shortest path is simply the connecting link between the grid point and a closest neighbor on a contour (Figure 5). If there are several closest neighbors, an arbitrary choice is made. The next step is to examine neighbors of neighbors (Figure 6) and, subsequently, neighbors of neighbors of neighbors (Figure 7). Continuing in such a fashion, *each new neighbor connects* by adding a link to a previously established shortest path, until the pattern shown in Figure 8 results.

The reader observes that the various shortest paths merge as they approach the contour line. In fact, this is an immediate consequence of the way they have been constructed: by linking points to previously constructed shortest paths.

An analogous pattern structure has been observed before in connection with drain patterns described in the Introduction (see Figure 3). There a drain pattern was characterized by assigning to each grid point between contours a successor grid point which realized the steepest down-slope. Here a successor grid point is a neighbor grid point closest to a contour. In both instances, we deal with patterns that are forests rooted at contours.

We turn to our demonstration. Figure 14 presents the contour lines which serve as inputs to our Mustang Mountain example. Figures 15 and 16 show the shortest path pattern generated from these contours.

We finally remark that, if there are two adjacent grid points whose shortest distances are to two different contours, then these two contours are adjacent to each other. The above distance determination thus also provides the means to identify the inter-contour regions within which drain patterns are determined as described in the following sections.

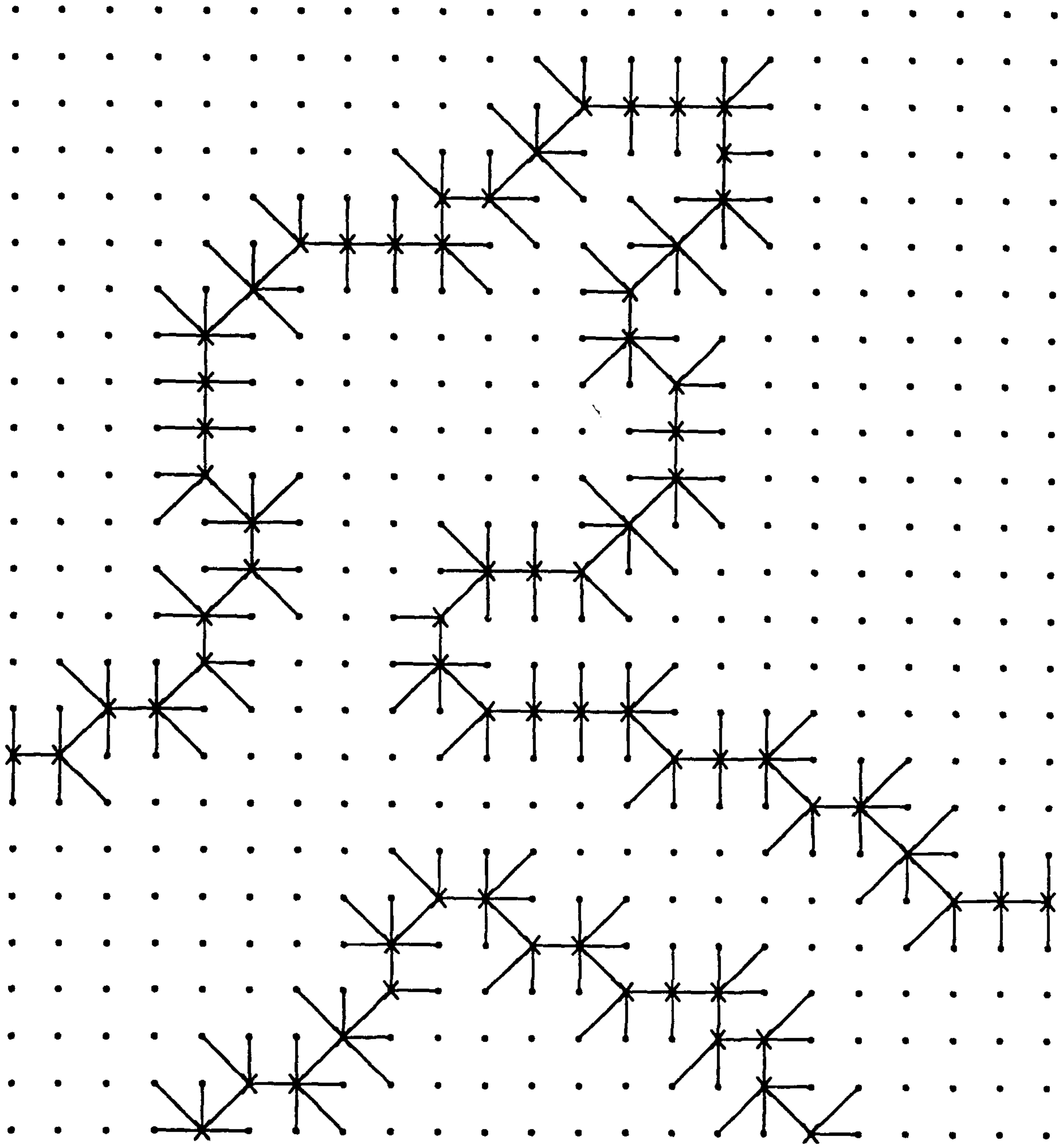

Figure 5: Grid neighbors of contours and their shortest connections: a first stage of determining a forest of shortest paths to contours.

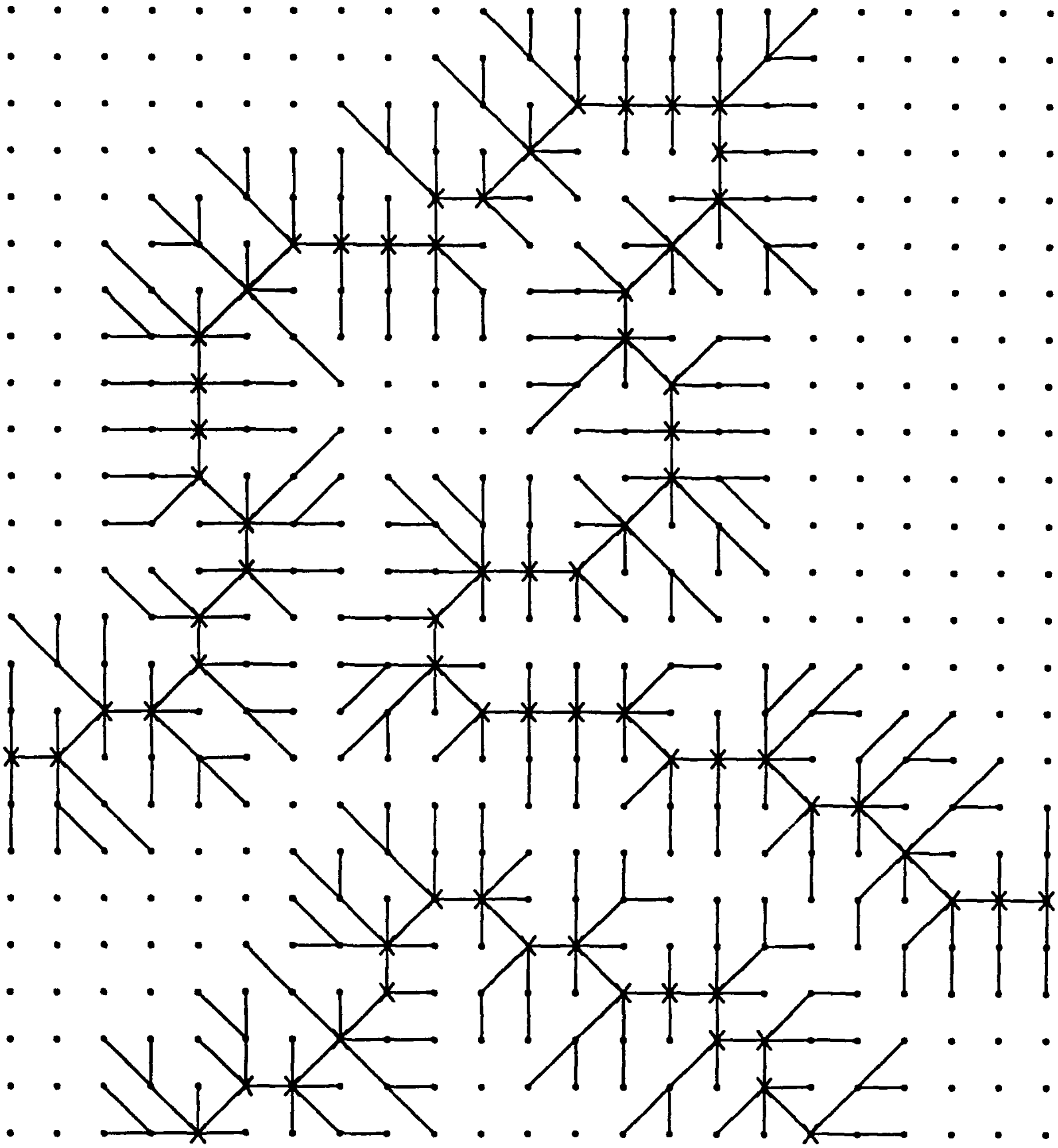

Figure 6: Grid neighbors of grid neighbors of contours and their shortest connections: a second stage of determining a forest of shortest paths to contours.

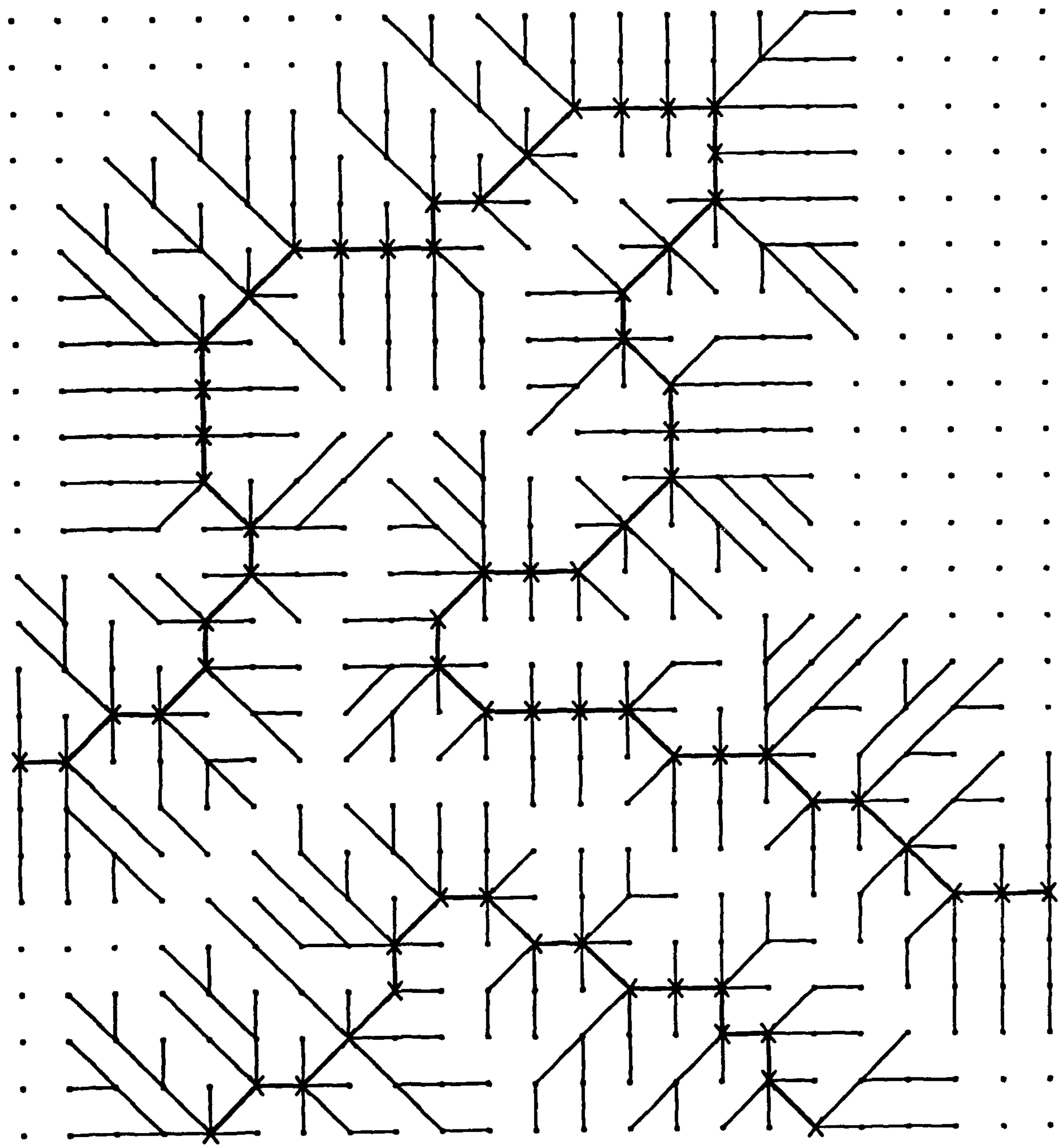

Figure 7: All shortest paths of three arcs: a third stage of determining a forest of shortest paths to contours.

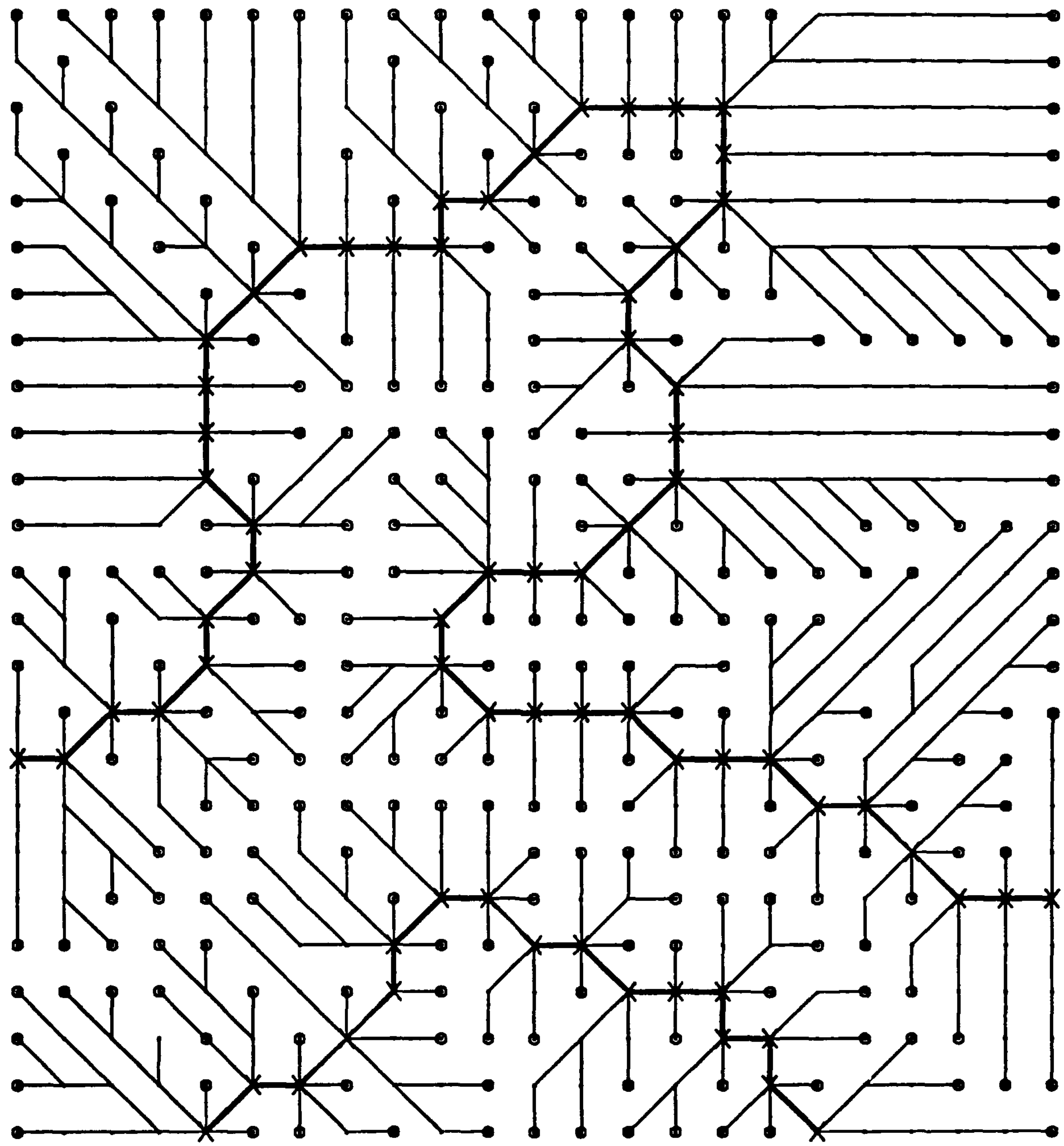

Figure 8: A completed forest of shortest paths to contours. Dead ends are marked with circles. The lengths of the grid paths towards their roots on a contour approximates actual euclidean distance.

3. Constructing a drain pattern from shortest paths

In this section, we discuss the generation of a complete drain pattern between an upper (source) contour and an adjacent lower (root) contour. We are guided in this endeavor by the obvious structural analogies between the drain pattern (Figure 3) discussed in the Introduction and the shortest path pattern (Figure 8) encountered in Section 2. Intuitively, drainage should proceed along the shortest possible routes towards the lower elevation. Also, such shortest paths merge into each other forming a forest, and a forest is the kind of pattern which we associate with drainage. However, working with actual shortest paths will not work for two reasons. First, if the upper contour is strongly curved – as it is in our example – the shortest paths from some grid points towards the lower contour may run into the upper contour, and thus leave the region between contours to which they are supposed to be confined. Second, for any surface, the direction of steepest descent at a point on a continuous contour line is perpendicular to that contour line. The path in the drain pattern should therefore also start out and end, at least approximately, perpendicular to the respective contours. Again, shortest paths based on the natural grid metric do not meet this requirement: minimizing distance as the "crow flies" does not take the shape of contour lines into account.

We therefore seek to modify the course of paths by making it expensive to linger close to contours. More precisely, we change the value of the distance between neighboring grid points so that it reflects their distance to contours. Let k_{i-1} and k_i be two distinct grid neighbors with natural grid distances $\mathrm{dist}(k_{i-1})$ and $\mathrm{dist}(k_i)$, respectively, from nearest contour points as determined in Section 2. Then we define their

$$\text{"adjusted distance"} \quad \mathrm{dadj}(k_{i-1}, k_i) \;=\; \frac{\mathrm{dist}(k_{i-1}, k_i)}{\sqrt{\mathrm{dist}(k_{i-1})} + \sqrt{\mathrm{dist}(k_i)}},$$

where $\mathrm{dist}(k_{i-1}, k_i)$ is the natural grid distance between two grid neighbors as set in Section 2: distance 1 for horizontal or vertical, and distance $\sqrt{2}$ for diagonal neighbors. In other words, the distance between neighboring points is weighted by the factor

$$\frac{1}{\sqrt{\mathrm{dist}(k_{i-1})} + \sqrt{\mathrm{dist}(k_i)}}.$$

This factor decreases with distance from contours. We say that an

"adjusted grid metric"

has been established as an alternative to the natural grid metric considered in the previous section.

For grid path $P(\underline{k}, \bar{k}) = \{\underline{k} = k_0, k_1, ..., k_n = \bar{k}\}$, the adjustment of the distance between grid neighbors gives rise to an

$$\text{“adjusted length”} \quad \sum_{i=1}^{n} \mathrm{dadj}(k_{i-1}, k_i).$$

In Section 2, we were interested mainly in distances of grid points from contours. The associated shortest path pattern was only a byproduct whose generation could have been suppressed. This situation is reversed at the present stage of our method. Given a region between adjacent contours, we are mainly interested in the pattern formed by paths of shortest adjusted length between grid points and the root contour; the adjusted distances themselves are a byproduct. Needless to say, the problem of determining a forest of paths of shortest adjusted lengths is again an instance of the classical shortest path problem in graphs with specified link lengths that was mentioned in Section 2.

It is now readily seen that any forest of paths of shortest adjusted lengths constitutes a complete drain pattern between two adjacent contours. Because movement close to any contour is penalized, any path of shortest adjusted length will tend to move towards a contour in the least expensive direction: perpendicular to the contour. For the same reason, it will not intersect a contour and will thus stay within its inter-contour region.

Note that the longest paths within the drain pattern represent major drain lines. This in itself is an important product of the method, since many surface generation methods can be drastically improved once major drain lines have been identified. Of course, only the location of such drain lines has been determined. If elevations along the drain lines are desired, then some prorating scheme needs to be employed. This will be the subject of the following section.

In our demonstration, Figures 17 and 18 show the drain pattern constructed by shortest paths with respect to the adjusted grid metric between the contours shown in Figure 14. Figure 19 marks the location of the main drainage line extracted from that drain pattern.

3.1 Ridge lines. The roles of the upper and lower of two adjacent contour lines can be reversed: consider the lower contour as the source contour and the upper contour as the root contour of the pattern of shortest paths with respect to the penalty-adjusted metric. The resulting pattern may be interpreted as a

“ridge pattern”.

Our method provides an option for such

"pattern reversal".

It can be used to determine ridge lines. A different application of this option will be demonstrated in Section 6.

In Figure 14 of the demonstration as well as in the example illustrated in Figures 1 and 3, each pair of adjacent contours, along with the grid boundary, delineates a distinct region of the grid. Those regions are well-defined and do not overlap. This observation is important because, in practice, contour lines are often incompletely drawn; they may show gaps or end prematurely. At this point of development, such cases are still beyond the scope of our method.

Finally we note that there are many other ways of defining grid metrics which penalize proximity to contours. For instance, instead of the sum of the square roots of the distances, one might have chosen the sum of the distances or the square root of that sum. We plan to discuss the properties of these various choices in a separate report, along with the question of suitable tie breakers in case paths of shortest adjusted distance are not unique.

4. Prorating elevations along a drain pattern

Consider a drain pattern in the region between two adjacent contours, and a drain path $P(k_1, r)$ from a dead end k_1, adjacent to a source contour point s, to its root r in the root contour. Connecting the source contour point s to its grid neighbor k_1 defines a grid path $P(s,r)$ from source to root contour. Since the elevations of both endpoints, $s = k_0$ and $r = k_n$, are known, the elevation at any intermediate point k_i, $0 < i < n$ of the

"prorating path" $P(s,r)$

can be estimated by

"prorating"

the elevation difference with respect to the length traveled within the path. By this we mean the following procedure: if

$$\text{length}P(s,r) \quad \text{and} \quad \text{length}P(s,k_i),\ 0 < i < n,$$

denote the length of the entire path $P(s,r)$ and of its portion $P(s,k_i)$, respectively, then we define the elevation z_i at point k_i as

$$z_i \;=\; z_0 + \frac{\mathrm{length}P(s,k_i)}{\mathrm{length}P(s,r)}(z_n - z_0),$$

where z_0 and z_n are the elevations at points $s = k_0$ (source contour) and $r = k_n$ (root contour), respectively. Grid point s, the starting point of the prorating path will be termed the

"contact point"

for the the dead end k_1.

We wish to extend that estimation procedure to all grid points in the drain pattern.

For this purpose, we have to deal with two additional cases. First, a grid point may lie on several different drain paths which stretch to the source contour. In that case different elevation estimates might be derived for that point depending on which one of the drain paths is selected for prorating. Second, not every drain path stretches to the source contour. In other words, the drain pattern may contain dead ends which are not adjacent to the source contour.

4.1 Partial prorating. The first ambiguity is resolved on a first-come, first-served basis that leaves previously set elevations in place. Suppose k_1 is a dead end with contact point s in the upper contour, and suppose no elevation had yet been set for k_1 during the prorating process. Then the drain path $P(k_1,r)$ connects the dead end k_1 in the drain pattern to a root r on the lower contour. It may be that this path contains no points whose elevation has been prorated so far. In this case, the root r is the only point for which the elevation is known, and prorating proceeds as described above. It may also be that the drain path $P(k_1,r)$ contains points whose elevation has already been prorated (see Figure 9). In that case, let $\bar{k}$ be the one among these points that – in the drain path – is closest to the dead end k_1. The point $\bar{k}$ splits the drain path $P(k_1,r)$ into the paths $P(k_1,\bar{k})$ and $P(\bar{k},r)$. The prorating procedure is such that since the elevation point $\bar{k}$ has already been determined, that holds true for all its successor points in path $P(\bar{k},r)$. In path $P(k_1,\bar{k})$, on the other hand, point $\bar{k}$ is the only grid point for which an elevation is known. Elevations for the remaining points can now be prorated between the contact point $s = k_0$ and the point $\bar{k}$, following the procedure described above with $\bar{k}$ in the role of the root r. In other words, the grid path $P(s,r)$ that stretches from the contact point $s = k_0$ to the root $r = k_n$ of the drain path of dead end k_1, may only be a

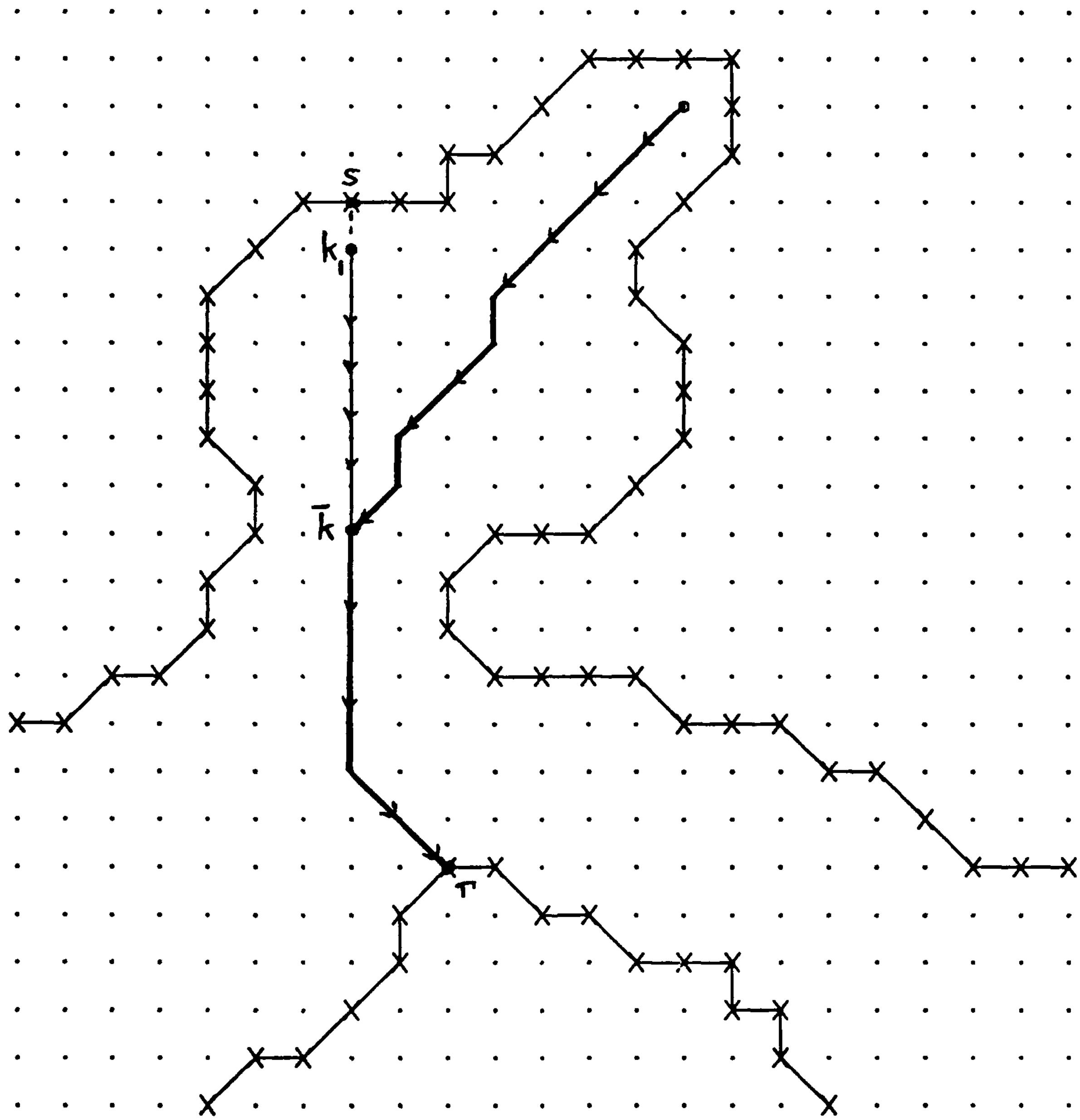

Figure 9: The elevations on the drain path indicated by the heavy line have already been set. Dead end k_1 of the drain pattern makes contact with grid point s on the source contour. Prorating takes place between s and the grid point $\bar{k}$, the point of set elevation closest to s in potential prorating path $P(s,r)$.

"potential prorating path",

only an upper portion of which will actually be prorated.

4.2 Contact criterion. Concerning the second difficulty, the drain pattern in Figure 3 exhibits many instances of dead ends that are not adjacent to their source contour, and have so far been excluded from the prorating procedure. In order to extend that procedure to such dead ends, we need to specify contact points other than those in the source contour. Of course, elevations for such contact points must have been already determined.

The proper choice of a

contact criterion,

that is, a criterion for the selection of contact points, turned out to be a major difficulty in the development of our method. The following criterion has been used for the demonstration in this report. However, it is subject to future adjustment, and we recommend against finalizing it in its present form.

Let grid point k_2 denote the successor of dead end k_1 in the drain pattern (see Figure 10), and let k_0 be a grid neighbor of dead end k_1. If k_0 has already been assigned an elevation and if it lies in the same region as dead end k_1 but does not belong to the root contour, then k_0 is a potential contact point for dead end k_1. We then call the angle (absolute value) between the direction from k_1 to k_0 and the ray extending from k_2 to k_1 the

"contact angle"

at dead end k_1. For potential grid point k_0 to be eligible as a contact for dead end k_1, we require that the contact angle does not exceed 135°. Furthermore, for a contact angle of 135° to be acceptable, k_1 must lie on the boundary of the grid; for a contact angle above 45° to be acceptable, k_0 must either be a point in the source contour, or the length of its drain path must exceed the length of that of k_1. Among all eligible contact points we choose the one that minimizes the contact angle. No tie breaker is currently in effect because of the preliminary nature of the criterion.

There is a – mainly conceptual – drawback to the above contact criterion. In order to describe this drawback conveniently, we assume for the moment that the upper contour serves as the source contour. Then our present contact criterion does not assure that the prorated elevations will decrease along descending drain paths. Indeed, consider a dead end k_1 not in contact with the upper (source) contour. Its contact point k_0 has consequently a lower elevation than the upper contour. Prorating takes place along a path between the contact point k_0 and a grid point $\bar{k}$ that is the first descendant of k_1 in the drain pattern for which an elevation has already been determined. This point $\bar{k}$ may under certain circumstances have a higher elevation than the contact point k_0, and prorated elevations may consequently increase along the drain path descending from the dead end k_1 to the grid point $\bar{k}$. This effect was observed only in rare instances and involved very small differences in elevation. Nevertheless, this conceptual flaw is one of several reasons why the current contact criterion should not be finalized.

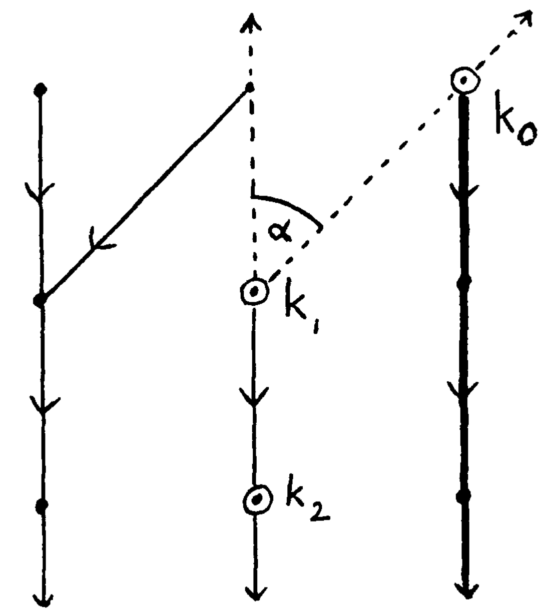

Figure 10: Grid neighbors of dead end k_1. The heavy line indicates a portion of the drain pattern with set elevations. The contact point k_0 is reached with the contact angle $\alpha = 45°$.

4.3 Elevation iterations. It is clear that the prorating procedure depends on the sequence in which the dead ends are examined. That examination proceeds in several

"elevation iterations".

During the first elevation iteration, each dead end is examined, in arbitrary sequence, to determine whether it has an eligible contact point on its source contour. If the answer is positive, such a contact point is selected according to the contact criterion, and associated with the dead end in question. Those dead ends with contact points are then sorted by the lengths of their drain paths. The dead end k_1 with the longest potential prorating path is processed first, followed by the dead end with the second longest path, and so on, in order of decreasing path lengths.

During the second elevation iteration, each nonprocessed dead end is examined again, this time for eligible contacts not in a source contour, that is, grid points which lie on paths processed during the first iteration and whose elevation has thus been determined. Best contacts are associated with their respective dead ends. These are again sorted and processed by decreasing lengths of their potential prorating paths. This process is repeated until no new eligible dead ends can be located. Between 50 and 100 elevation iterations are typically encountered for a landscape such as the portion of the Mustang Mountain area used for our demonstration. We stress again that this procedure along with the criteria for contact is the subject of ongoing research.

Not all grid points may be assigned an elevation in this fashion even if they are contained in a complete drain pattern. Consider a region enclosed by a highest "last" closed contour, that is, a region containing neither additional contours nor spot elevations, thus being bounded only by that one closed contour, as in the case of a peak for which no summit elevation is spotted. Selecting such a closed contour as root contour, we can construct in its interior a pattern of shortest paths with respect to the penalty-adjusted metric (it can be shown to coincide with the natural shortest path pattern). This pattern can be interpreted again as a drain pattern, but there is no contour to serve as source contour. Consequently, prorating cannot take place. For a similar effect to occur, the bounding contour need not be fully closed, as the example in Figure 11 shows. We are presently investigating the use of slope information in order to assign elevations in peak regions enclosed by last contours.

4.4 Descent restriction. The drain pattern of Figure 3 yields a drain line associated with a valley. Prorating along such a drain line in strictly linear fashion, however, was found to yield unsatisfactory results. Indeed, moving in the proximity of a contour line and parallel to it is generally expected to result in little change of elevation. Similarly, approaching a contour from below is associated with ascent. At some portions of the aforementioned drain

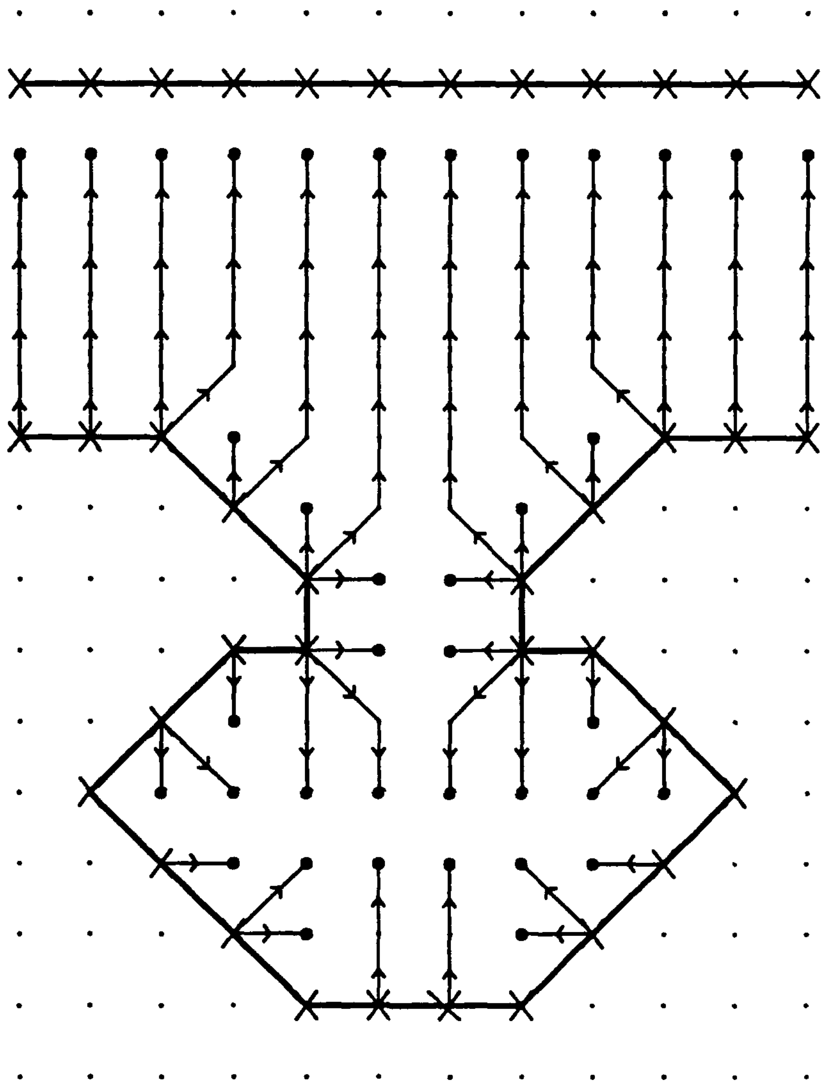

Figure 11: A complete drain pattern between a root contour below and a source contour above. The dead ends are marked by circles. Note that the dead ends in the lower portion of the pattern do not make contact with prorated elevations at any time of the prorating process. This provides an example for the fact that the setting of elevations by the prorating process may not reach all grid points between two adjacent contours.

line, the drain line moves closer to the surrounding source contour before moving away again. Figure 12 shows the drain line extracted from the drain pattern in Figure 3. Its dashed arcs indicate the portion of the line along which the source contour is approached. We feel that along such portions of the drain line, the rate of descent should be lower than indicated by the linear prorating scheme. The same should hold for any drain path in the pattern. We therefore adhere to the following

"descent restriction":

when proceeding along a drain path in the direction of its root results in reducing the natural distance to the source contour, then prorating of elevations is suspended for those portions of the drain path along which such distance reduction occurs. (In the case of a reversed pattern, the term "ascent restriction" would be more appropriate.) Strictly speaking that means that certain portions of the drain path are kept horizontal while the remainder of the path is subject to linear prorating. In our present implementation, however, we retain a very small nominal rate of descent in those portions that would otherwise be flat.

According to our experience so far, the descent restriction definitely improves the quality of the generated surface.

5. Pattern iteration

Consider a region between two contours for which elevations can be fully determined by prorating, and assume that the corresponding region of a grid elevation matrix has been determined from the initial drain pattern. Then – again in that region – a drain pattern derives from the elevation matrix by assigning to each grid point a neighboring successor point on the basis of steepest descent. If the resulting drain pattern is complete, then it can again be used for prorating. We call the process of alternating between prorating from a drain pattern, and deriving a drain pattern from an elevation matrix

"pattern iteration".

An important feature of a pattern iteration step is that it does not change the main drain lines in any essential way. This can be established theoretically and is borne out by the demonstration results. Thus the new drain pattern is again meaningful. Is it improved? We contend the answer is positive. For justification, we refer to the results of the demonstration. In any case, pattern iteration is an option available within the general approach to surface generation based on drain patterns.

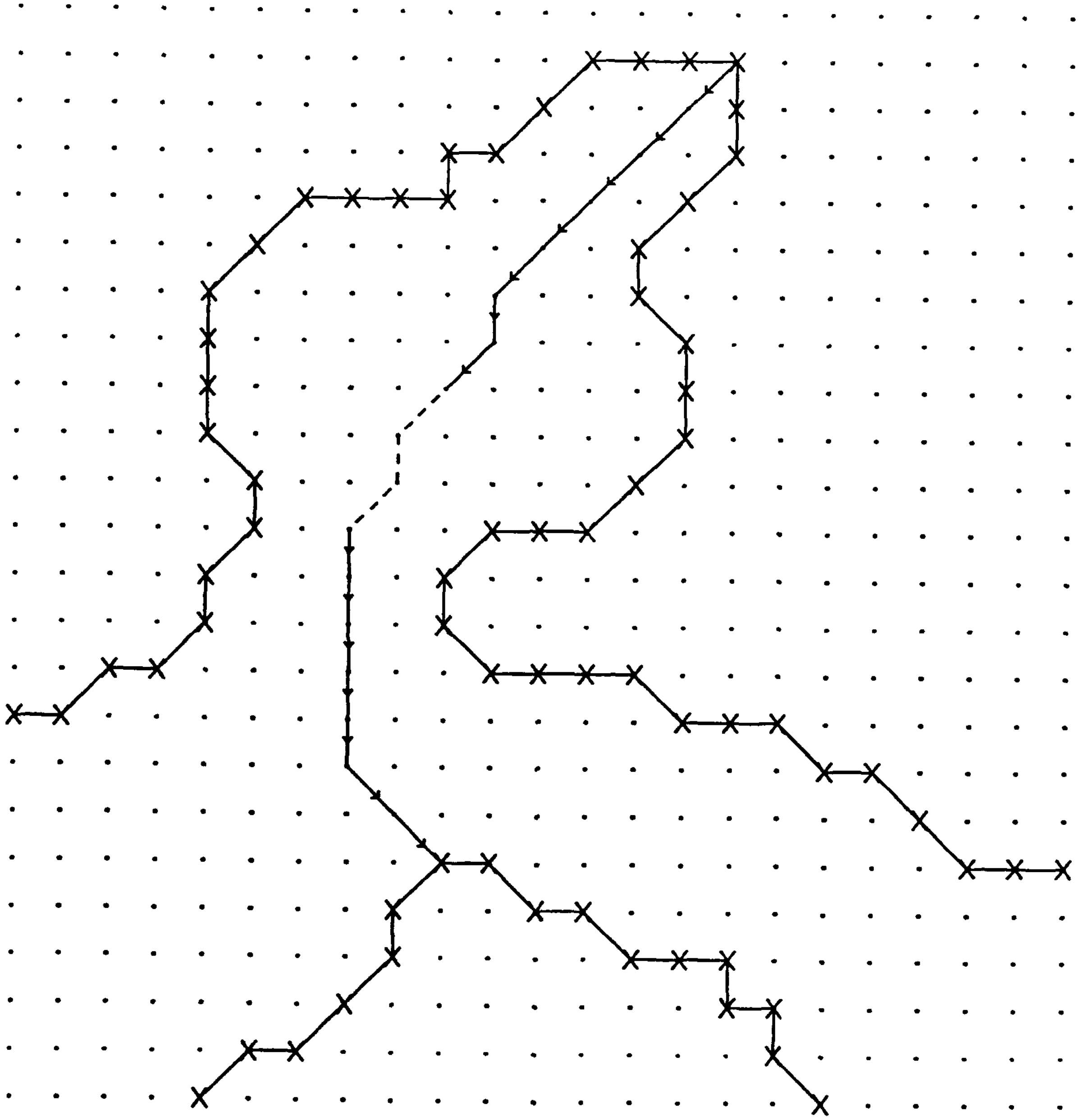

Figure 12: Drain line with portion (dashed) along which descend restriction takes place during the elevation prorating process.

Does pattern iteration converge? The answer is usually negative, because linear prorating of elevations is too simplistic in that it is only based on length – with the exception of prorating under the descent restriction – and ignores the drain information at neighboring points. We suggest therefore to employ a process of successively

"averaging"

the previously generated elevations. By this we mean that an infinite sequence of positive

"weights" $w_1, w_2, \ldots$

is specified, and a sequence

$$Z^{(0)}, Z^{(1)}, Z^{(2)}, \ldots$$

of grid elevation matrices is defined by the recursion

$$Z^{(i)} = (1 - w_i)Z^{(i-1)} + w_i z^{(i)}, \quad i = 1, 2, \ldots,$$

where $Z^{(0)}$ is the elevation matrix arising from the initial drain pattern, and $z^{(i)}$ is the elevation matrix generated by prorating along the drain pattern of $Z^{(i-1)}$.

For our demonstration, we have used the weights

$$\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots .$$

The resulting process of averaging, generally called "exponential smoothing", can be shown to converge.

Averaging is attractive not only because it forces convergence but also because it breaks up the linearity of descent along some drain paths.

At this point, we should describe the process of deriving a drain pattern from a grid elevation matrix in more detail. Let k denote a grid point, and let k_i, $i = 1, \ldots, 8$, denote its eight grid neighbors with elevations z_i, $i = 1, \ldots, 8$. If the grid point k lies on the boundary of the grid, then fewer neighbors are available, but the following argument does not change significantly. If none of the neighbor elevations is strictly lower than the elevation z at grid point k, then there will be no successor to k in the drain pattern: grid point k is either the low point of a sink or belongs to a flat spot. If there are lower neighbors, then the (positive) down-slope from k to such a neighbor k_i is given by

$$\frac{z - z_i}{1} \quad \text{or} \quad \frac{z - z_i}{\sqrt{2}}$$

depending on whether k_i is a straight or a diagonal neighbor of k. A grid neighbor of largest positive down-slope is the selected successor of grid point k in the drain pattern. We do not use at present a tie-breaking rule. (An alternate procedure would be to select simply a lowest neighbor of k provided that it is lower than k. This procedure leads to drain paths that have a smoother appearance, because such paths may not have successive arcs of grid length 1 at right angles.)

5.1 Sinks and cross-over. As was stressed at the beginning of this section, the drain pattern derived from any elevation matrix determined during pattern iteration must be complete in order to permit subsequent prorating. Unfortunately, we cannot guarantee completeness because the averaging of elevation matrices may still generate sinks if, say, a ridge in one elevation matrix crosses a ridge in the other. In other words, the (weighted) sum of two drain-consistent elevation matrices need not be drain-consistent.

In order to arrive at a complete drain pattern, a

"sink removal"

procedure must be put in place. There are two options. To explain these options, we invoke the imagery of filling a sink with water until it overflows. The first option is to raise every grid elevation covered by water to overflow level. The second option requires identifying an overflow point, that is, a grid point whose elevation is at overflow level and one of whose neighbors does not drain into the sink in question. The drain path of this grid neighbor is then followed, until either elevation has fallen below the low point of the sink or the low point of another, necessarily lower, sink has been encountered. Elevations are then reduced linearly along that path, permitting the sink to drain at least into a lower sink if not all the way to a contour. This process is repeated until all sinks are removed. Similar procedures are available for dealing with flat spots. However, since floating point arithmetic is used for prorating elevations, it is extremely unlikely that grid neighbors will have exactly the same elevation.

Our present demonstration software uses the second option. The reader might be concerned that input data might be altered in this fashion. However, only prorated elevations are affected. Also, it has been our experience that elevation corrections during sink removal have typically involved only fractions of inches.

Early on in our experimentation with pattern iteration we stabilized the iteration by keeping the drain pattern fixed in the

"lower rim"

of the region between contours. The lower rim (more aptly called "upper rim" in case of pattern reversal) is defined as that portion of the previous drain pattern in which the drain paths are actual shortest paths to the root contour. At this point in time, we are not certain whether this stabilization is needed or even beneficial. We found that it caused one irregularity, namely,

"cross-over"

in the new drain pattern. By this we mean the occurrence of both diagonals in a unit cell as arcs in the same drain pattern (Figure 13). If the successors in the drain pattern are selected on the basis of steepest down-slope or, in case of pattern reversal, up-slope, then it is easily seen that cross-over is ruled out. However, at the juncture between the variable drain pattern and its fixed portion in the lower rim, that selection rule is suspended, and cross-over may occur in isolated instances. Our implementation tests for and corrects such instances.

The results of the first iteration are displayed in Figures 20 and 21, the drain pattern after four subsequent iterations is shown in Figures 22 and 23. Pattern iteration is stopped after the fifth elevation matrix has been determined. The two last matrices in the sequence differ only by inches in their elevations. The reader also notices that the display of the second drain pattern in Figure 20 already appears to illustrate the "lay of the land", a visual effect absent in the initial drain pattern.

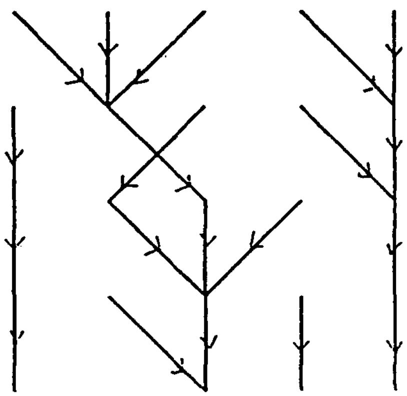

Figure 13: Cross-over in a drain pattern.

6. Interspersing contours

For some applications and for support of other terrain modeling methods, contour lines at elevations between the elevations of given adjacent contour lines are of primary interest. We will now demonstrate that the techniques described previously in this report can indeed be utilized to generate such intermediate contours.

Software developed during the course of this project can be used to extract contour lines at any specified integer elevations from the grid elevation matrix generated in Section 5. In order to become grid-digitized, points on these contour lines are reduced to closest grid points even if the elevations estimated previously at those grid points deviate from the elevation specified for the contour.

In this fashion, we can therefore

"intersperse"

a grid-digitized contour at any prescribed integer elevation between the elevations of the upper and the lower contour, respectively. Selectively interspersing contours may prove to be an important tool for upgrading the performance of other terrain surface generation methods such as TIN based methods, supplementing the capability to determine major drain lines.

We also observed in previous experiments that the quality of an interspersed contour is best at an elevation close to its source contour. This suggested the first step for our demonstration: use the final elevation matrix generated from contours at 4500 ft, 4525 ft, and 4550 ft to intersperse two contours at 4520 ft and 4545 ft, respectively (Figure 24).

The distances between adjacent contours including the new ones are less than the distances between the original contours. Since the newly interspersed contours are good contours, the reduced distances lead to improved prorating. We chose drain reversal for prorating between the five contours of the appended set (Figures 25 through 27). Now contours close to the respective lower contours, namely, at 4505 ft and 4530 ft were extracted and adjoined to the previous five contours (Figure 28). The method was then used again, now with the normal drain orientation (Figure 29), for generating an elevation matrix from which contours at 4515 ft and 4540 ft were extracted. These contours were again adjoined to the previous ones (Figure 30), and the method repeated with pattern reversal (Figure 31). Contours at 4510 ft and 4535 ft were extracted from the resulting elevation matrix and interspersed between previously determined contours, providing a full set of contours at steps of five feet between the elevations of 4500 ft and 4550 ft (Figure 32). The final elevation matrix and drain pattern are then determined using normal drain orientation (Figures 33 and 34). The major drain line is displayed in Figure 35.

7. Summary

The report addresses (i) the extraction of geographic features and (ii) the interpolation of elevations from given contour lines and their elevations. A new approach is proposed, which may prove valuable as a stand-alone terrain modeling tool as well as in support of other techniques such as TIN methods.

To this end, a drain pattern that is conceptually analogous to a drainage network is constructed between each pair of adjacent contour lines. The construction presupposes a regular rectangular grid. The natural grid metric, which assigns the regular euclidean distance to each vertical, horizontal, and diagonal unit link, is adjusted to make links in the neighborhood of contour lines more expensive than those that are further away. Then the shortest path pattern from the upper contour to the lower contour has the conceptual properties of, and is interpreted as a first approximation to, the actual drainage network.

This first cut at constructing a drainage network already yields credible feature information in the form of major drain lines.

Elevations can be prorated along a given drain pattern. This yields a grid elevation matrix, which in turn can be used to derive a drain pattern. Elevations can again be prorated along the latter drain pattern, leading to an additional elevation matrix. Iterating that process and applying exponential smoothing to the resulting sequence of elevation matrices yields a more realistic drain pattern and concomitant improved elevation estimates.

Further feature information in the form of interspersed contour lines can be extracted. These processes are demonstrated for an example based on DMA data.

While the new approach shows promise, it needs further development in order to become a reliable tool for large scale terrain modeling.

Throughout this report, it was assumed that the underlying grid was composed of square unit cells. In some applications, such as those based on grids originally defined in terms of latitudes and longitudes, the unit cells of the underlying grid are not square. While all considerations in this report can be readily extended to cover those applications, present software is still restricted to square unit cells. Also, we do not expect our present software to be very sensitive to small deviations from squareness.

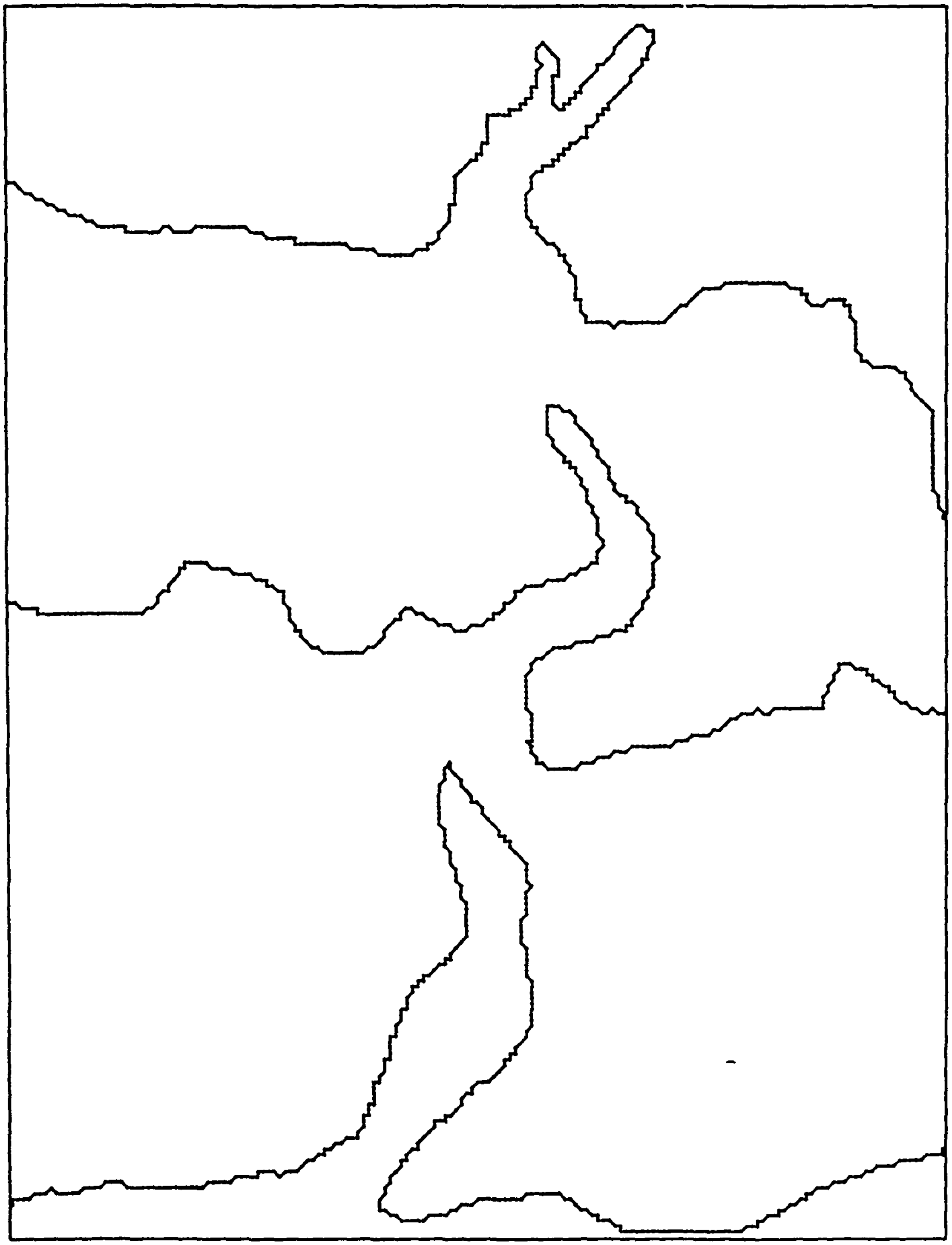

Figure 14: A 3400 by 4400ft map area located near Mustang Mountain, Fort Huahuca, Arizona, with three grid-digitized contours at elevations of 4500ft, 4525ft, and 4550ft, respectively. The grid unit is 20 by 20ft. The contour information shown here represents the sole input for generating the material displayed subsequently.

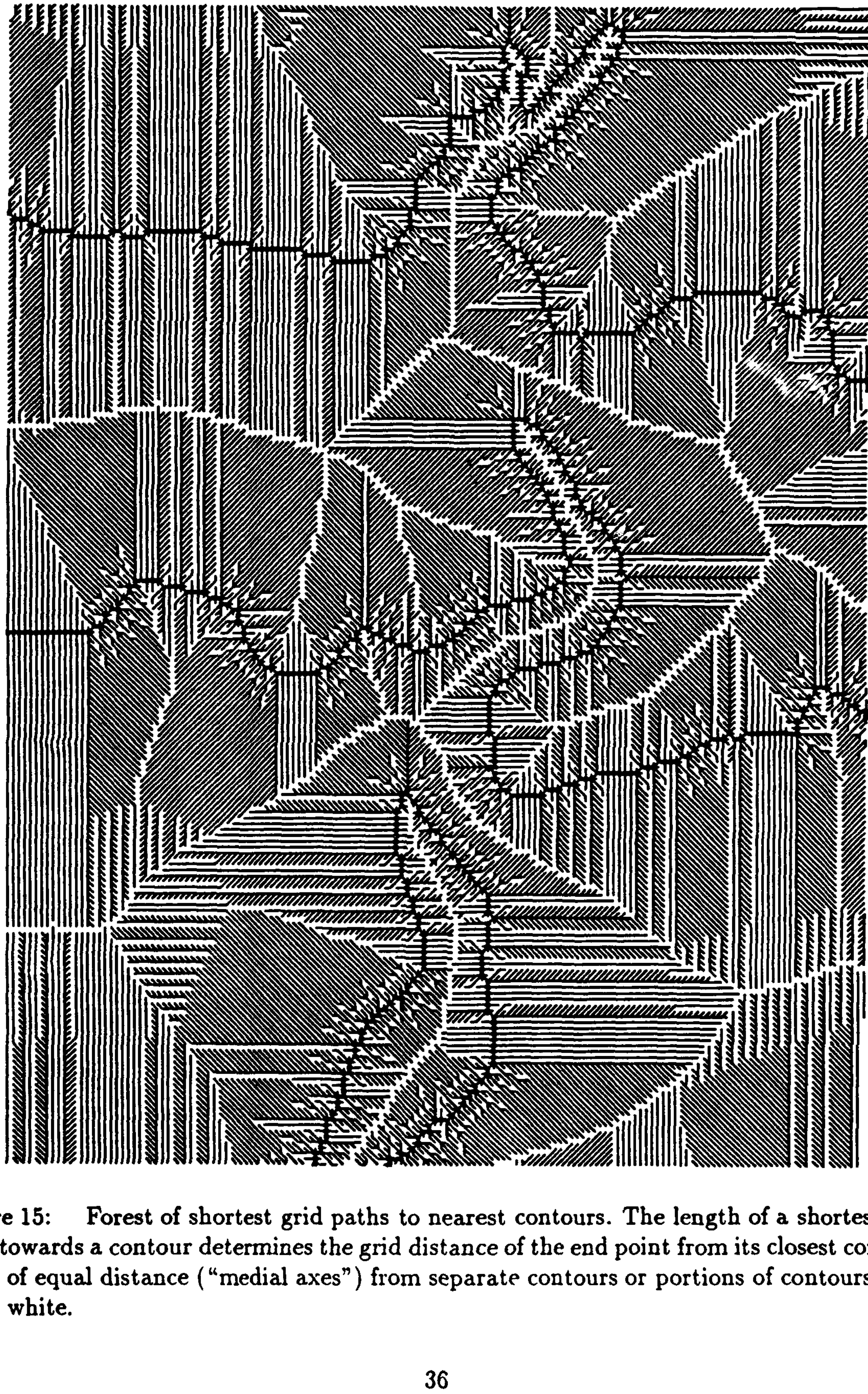

Figure 15: Forest of shortest grid paths to nearest contours. The length of a shortest grid path towards a contour determines the grid distance of the end point from its closest contour. Lines of equal distance ("medial axes") from separate contours or portions of contours show up in white.

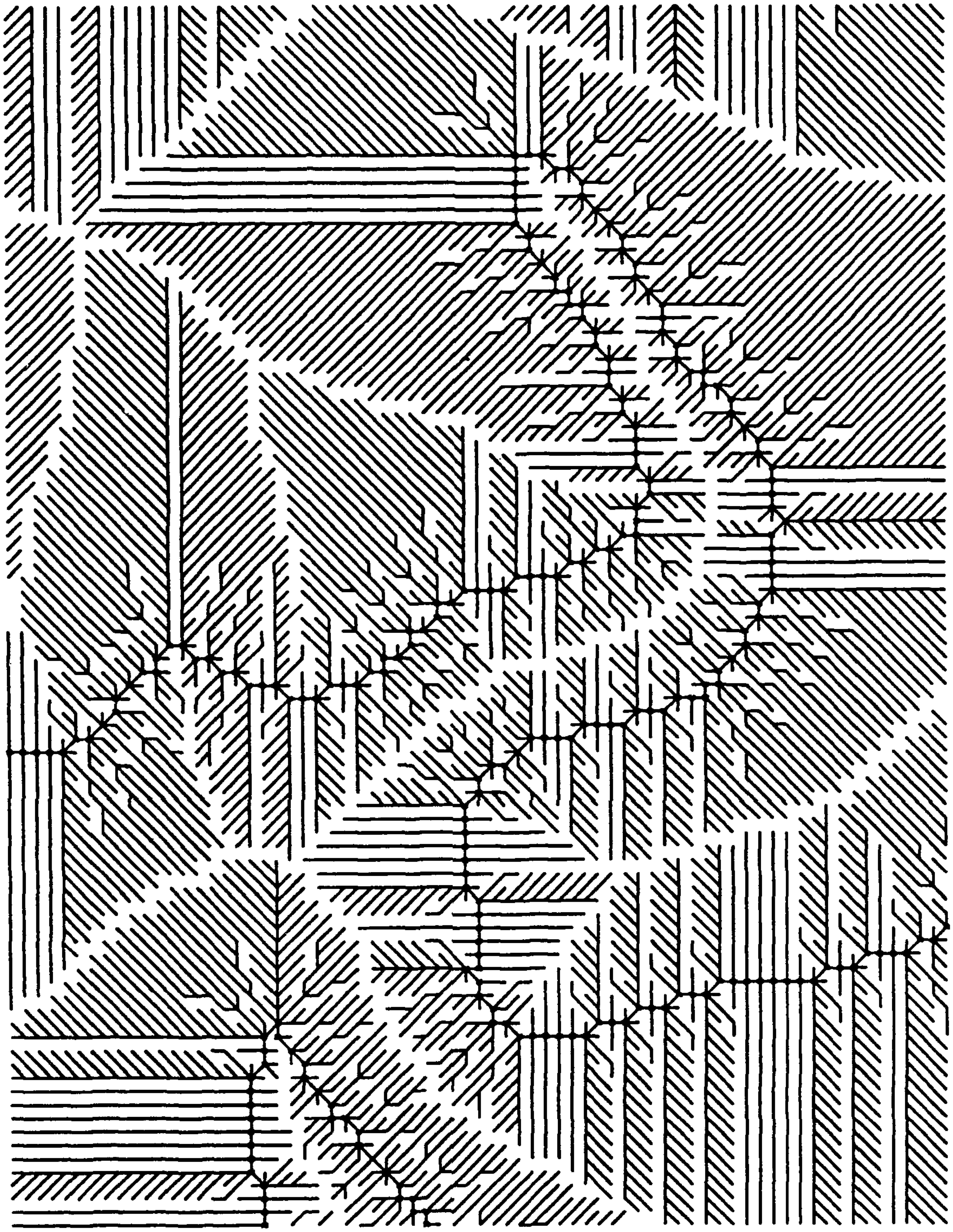

Figure 16: Enlarged portion of Figure 15 Distance values derived from such patterns provide a distance weighing scheme to be used for generating a first drain pattern.

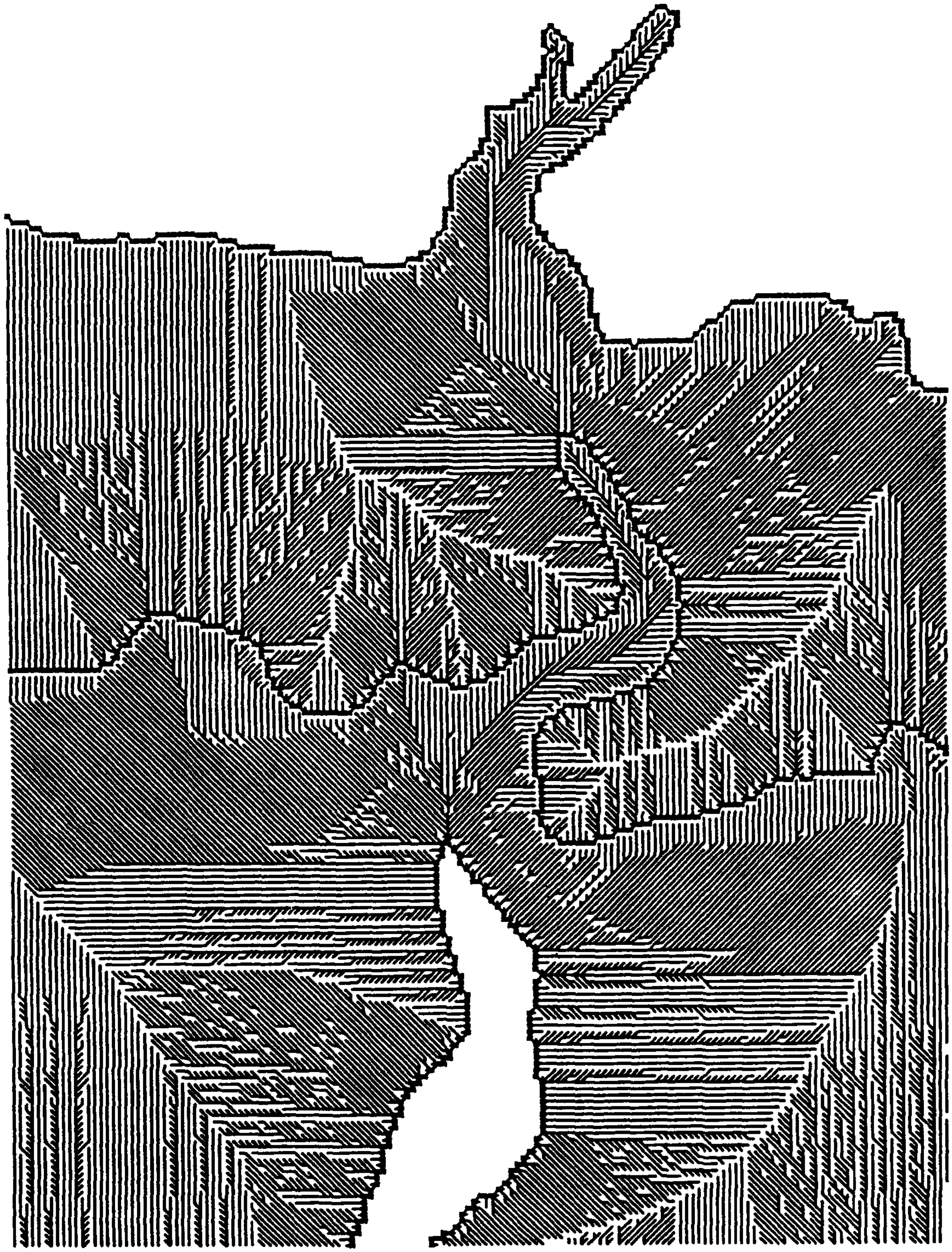

Figure 17: Drain pattern generated from penalty-adjusted shortest paths between the contours at 4500ft, 4525ft, and 4550ft displayed in Figure 14. The drain pattern is complete in that every drain path eventually leads to a contour point.

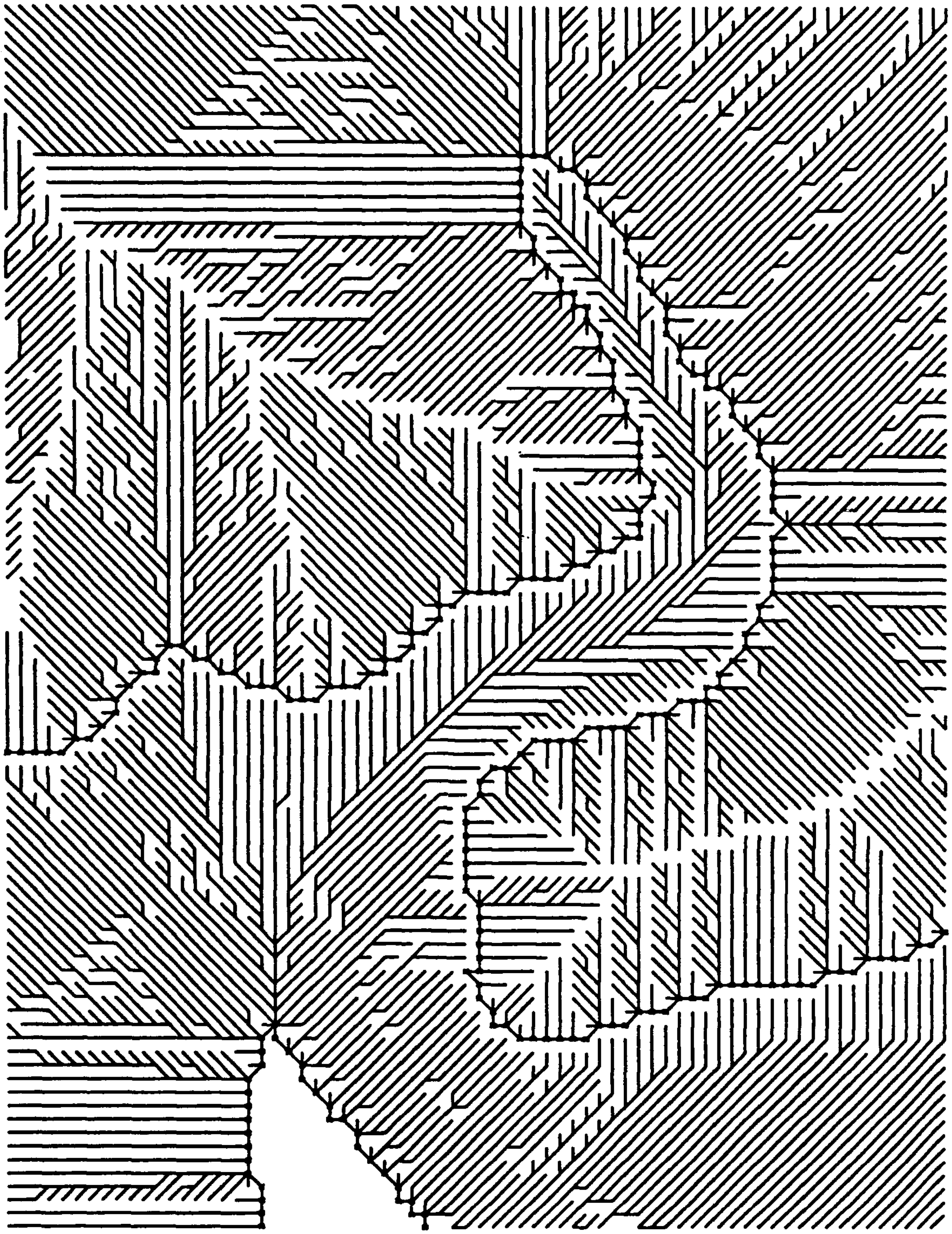

Figure 18: Enlarged portion of Figure 17. The drain pattern will be used to prorate elevations.

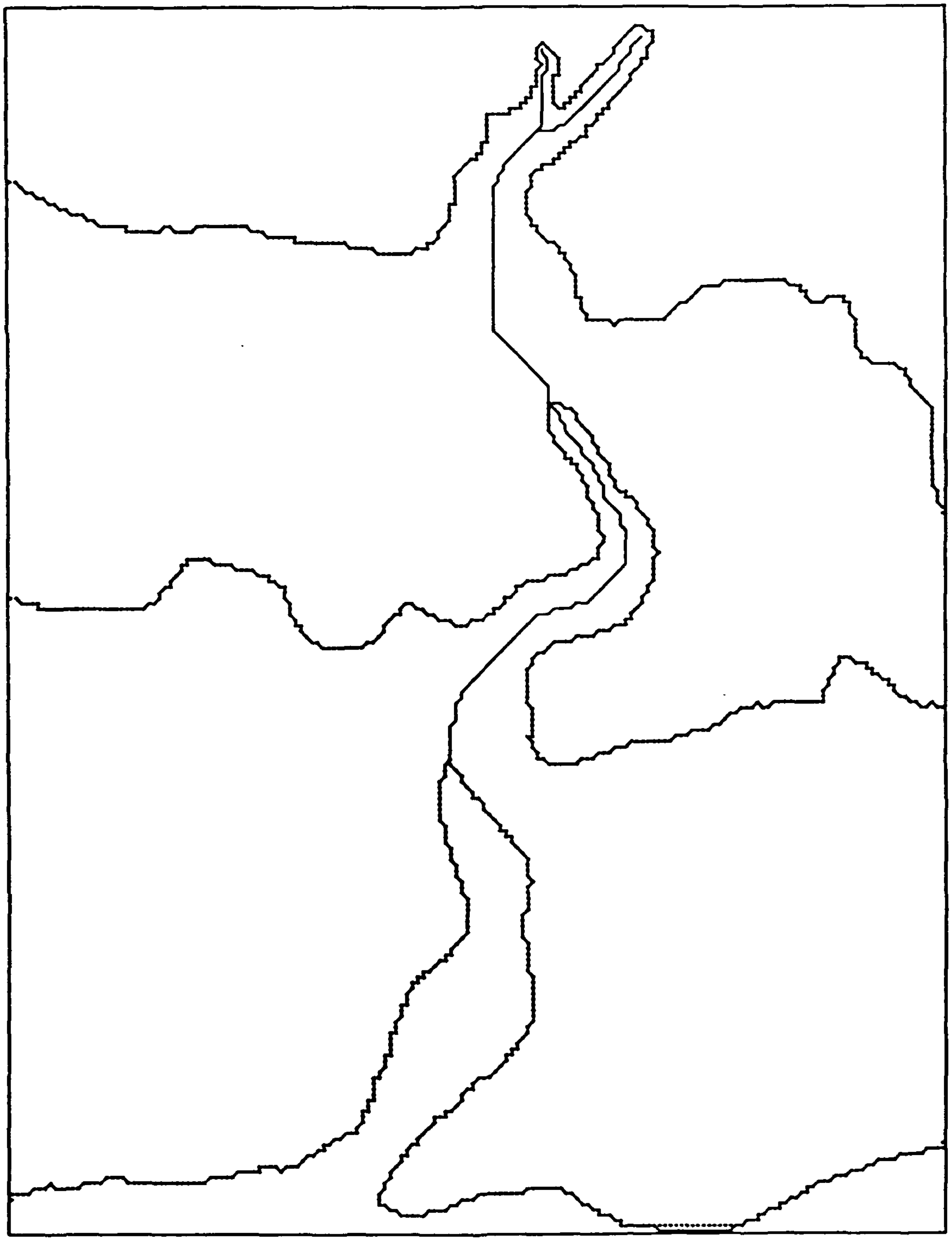

Figure 19: The three original contours with major drainage line extracted from drain pattern shown in Figure 17. This drain pattern and, consequently, the drainage line are derived solely on the basis of shortest paths techniques from the contour lines shown in Figure 14.

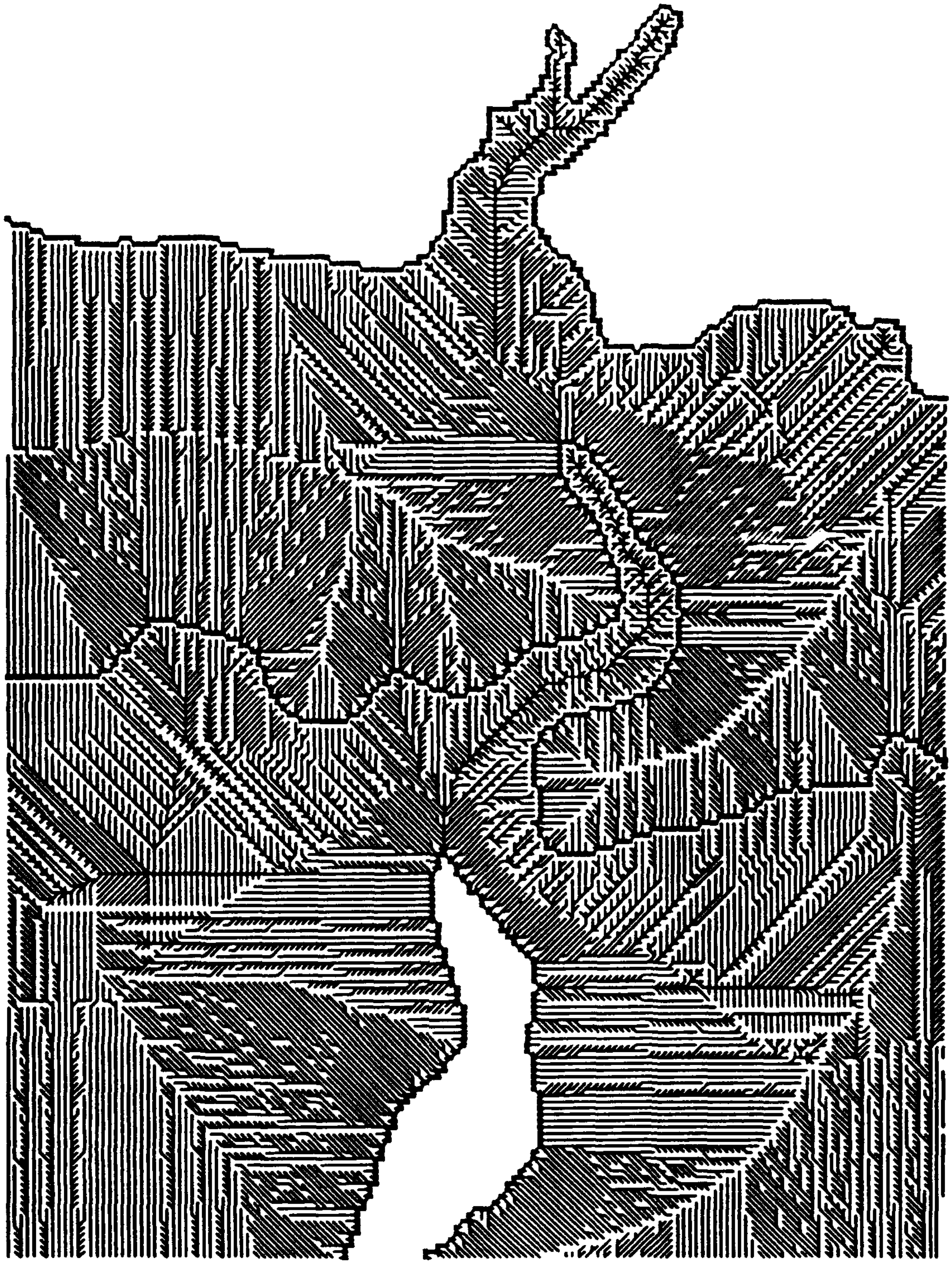

Figure 20: Second drain pattern in pattern iteration. This drain pattern derives from elevations that were prorated following the initial drain pattern shown in Figures 17 and 18. This drain pattern starts to look more realistic. It will again be used to prorate elevations which will be averaged with the previous ones.

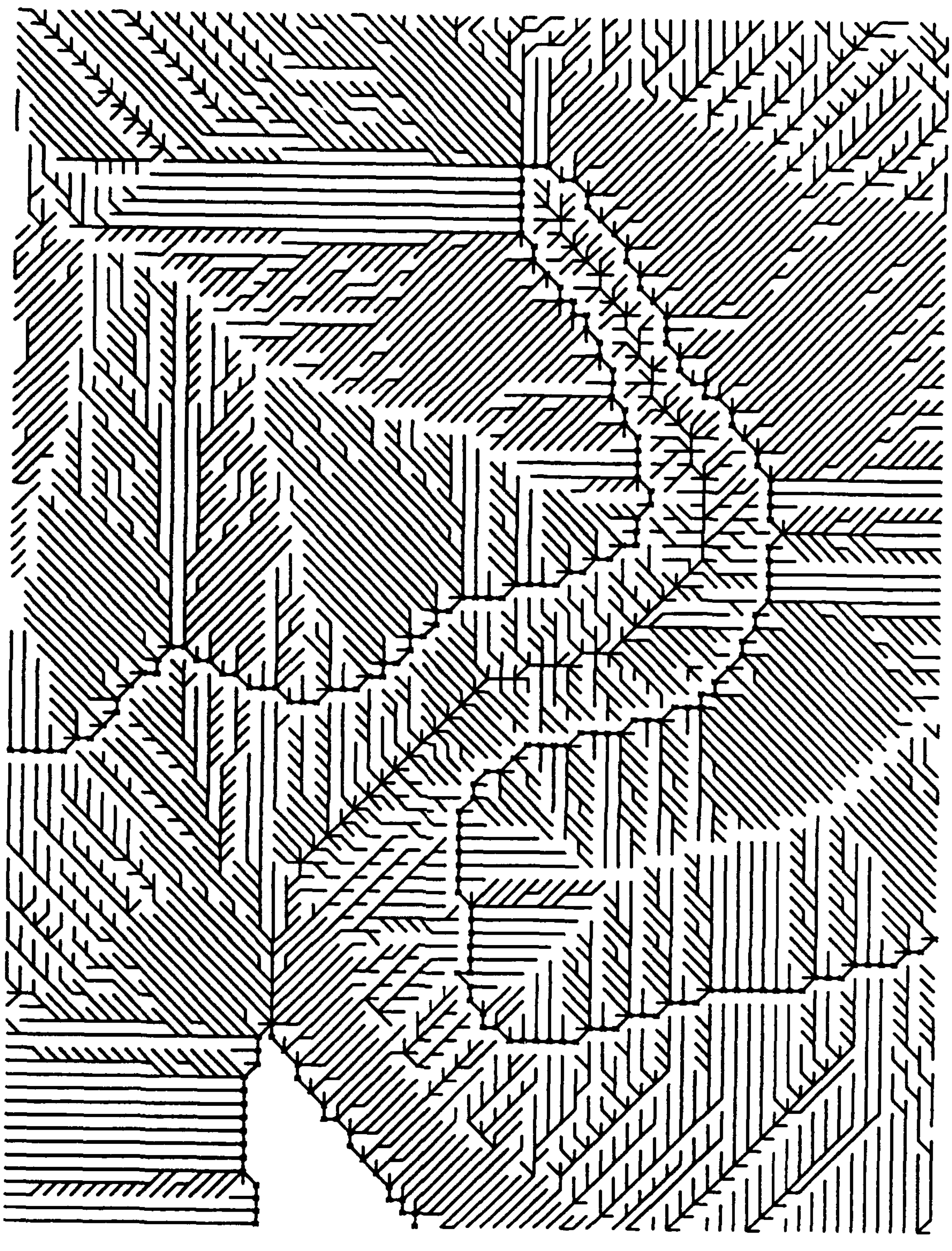

Figure 21: Enlarged area of Figure 20. Note that the main drainage line has remained unchanged, but is more apparent visually than in Figures 17 and 18.

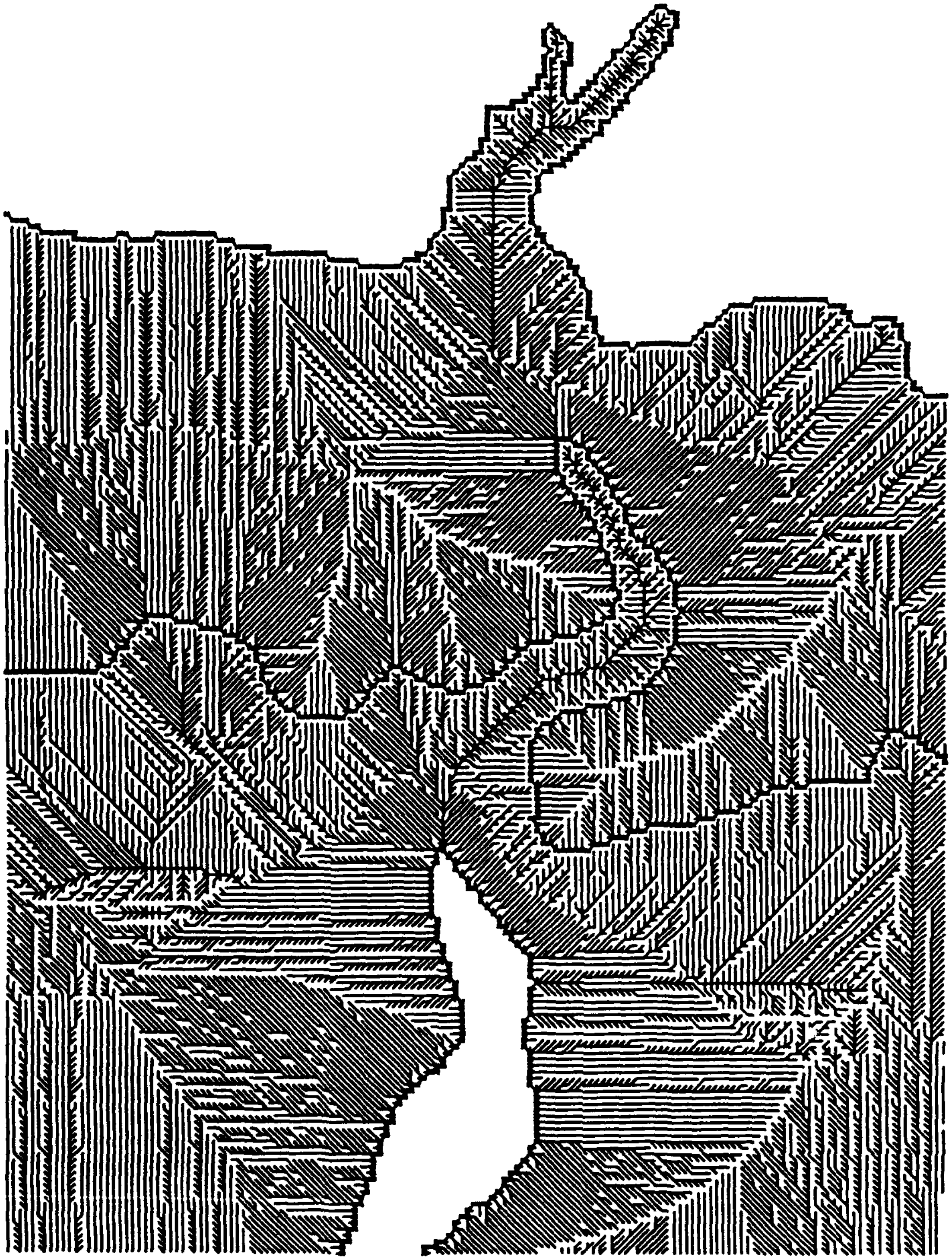

Figure 22: Drain pattern after five iterations, each of which consisted of prorating elevations and deriving their drain pattern.

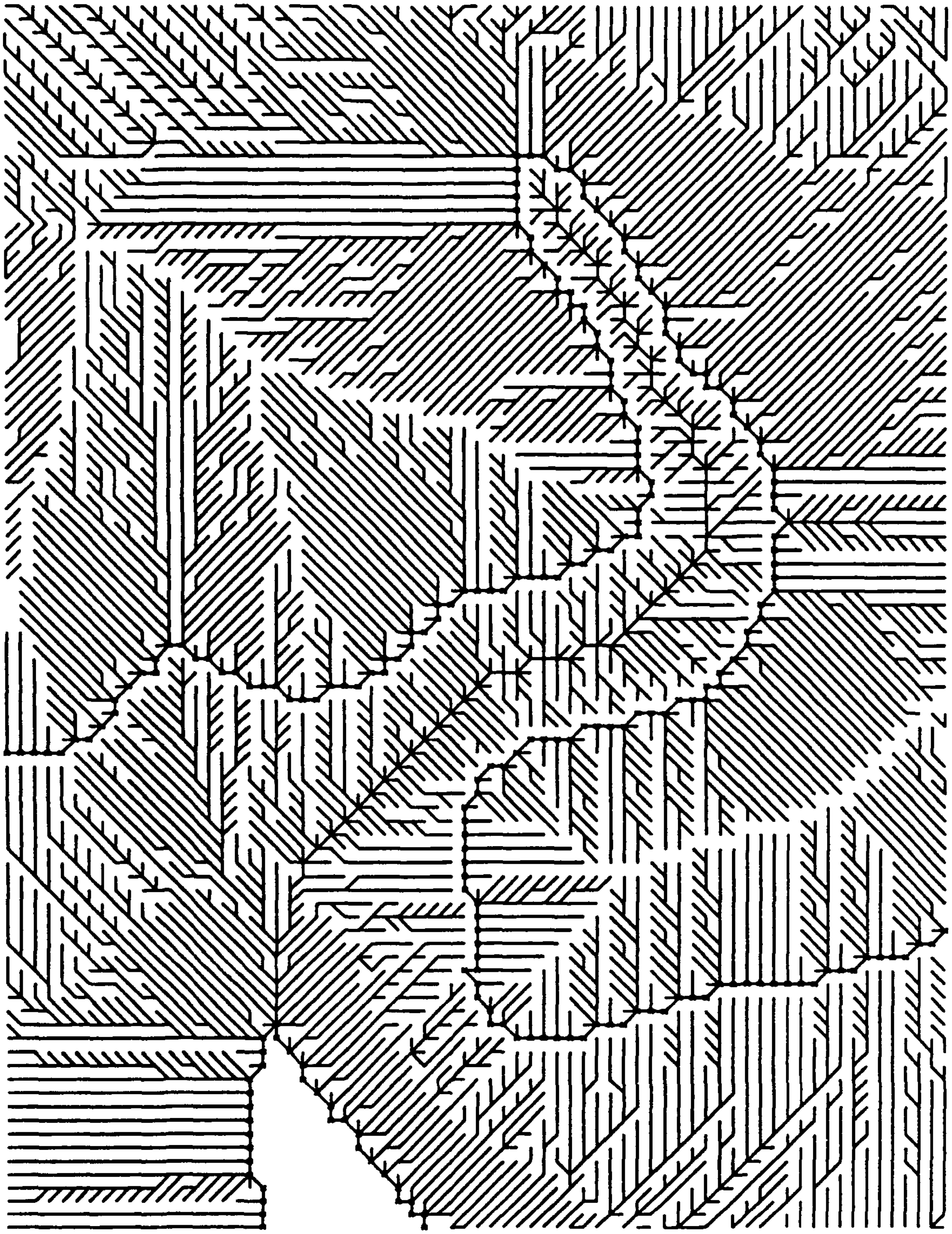

Figure 23: Enlarged portion of Figure 22. The main drainage line is still unchanged. The iteration was stopped with this drain pattern after its prorated elevations had been averaged into the previous average.

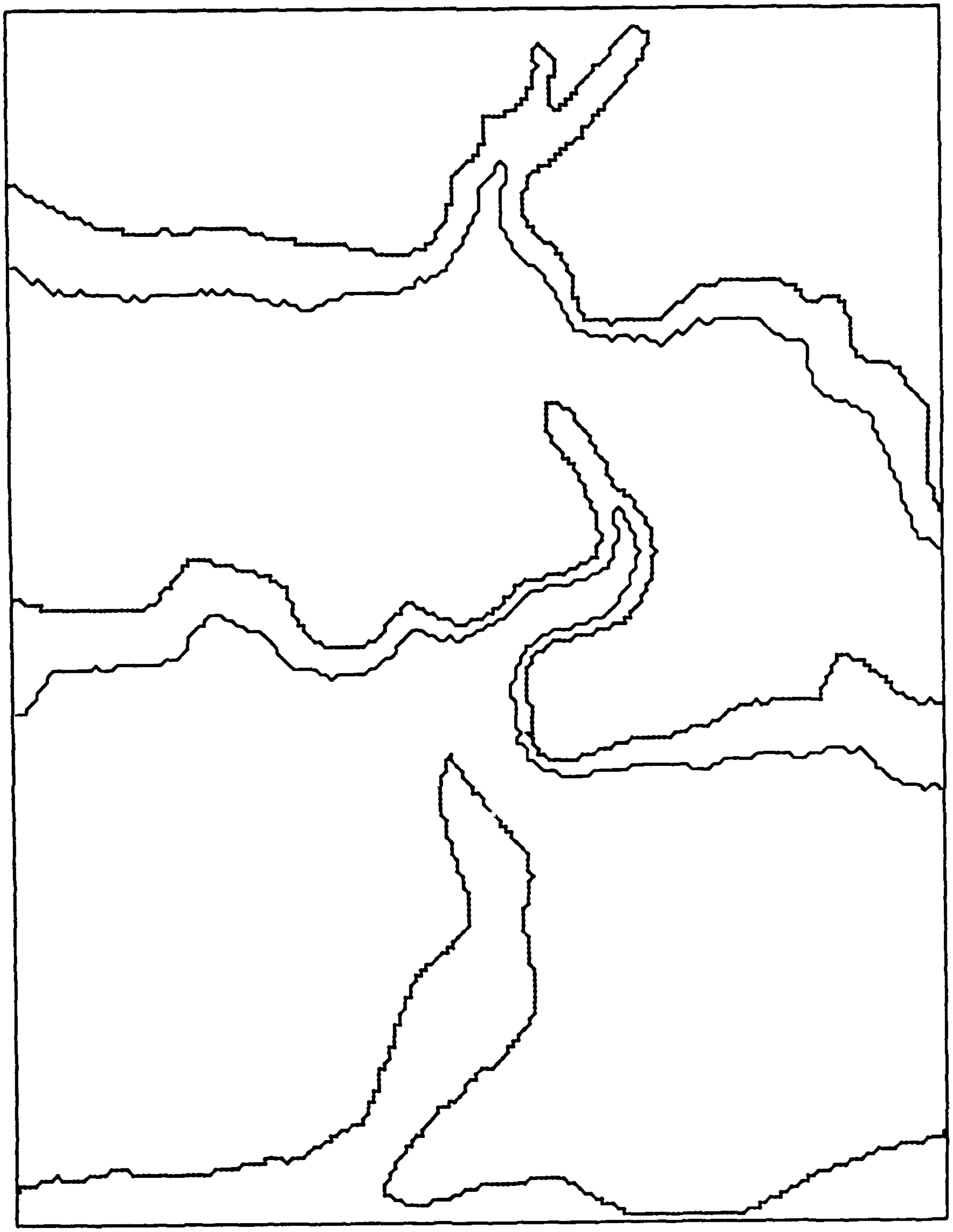

Figure 24: Two contour lines at 4520ft and 4545ft have been interspersed between the initial contour lines at 4500ft, 4525ft, and 4550ft. The new contours were extracted from the elevation matrix generated by the previous pattern iteration.

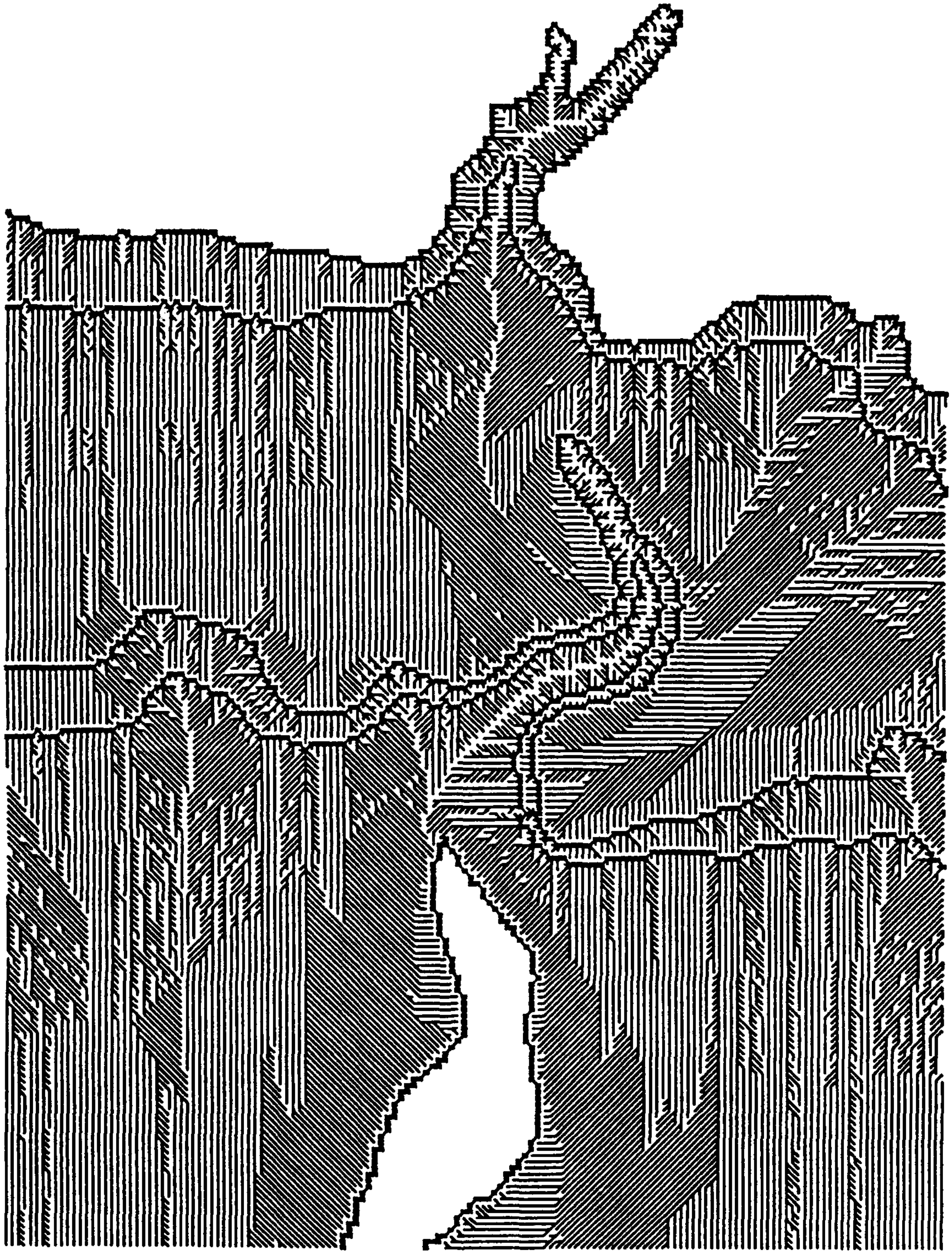

Figure 25: The initial drain pattern generated from the contour lines at 4500ft, 4520ft, 4525ft, 4545ft, and 4500ft using pattern reversal. Deriving this drain pattern initiates the second step of elevation generation.

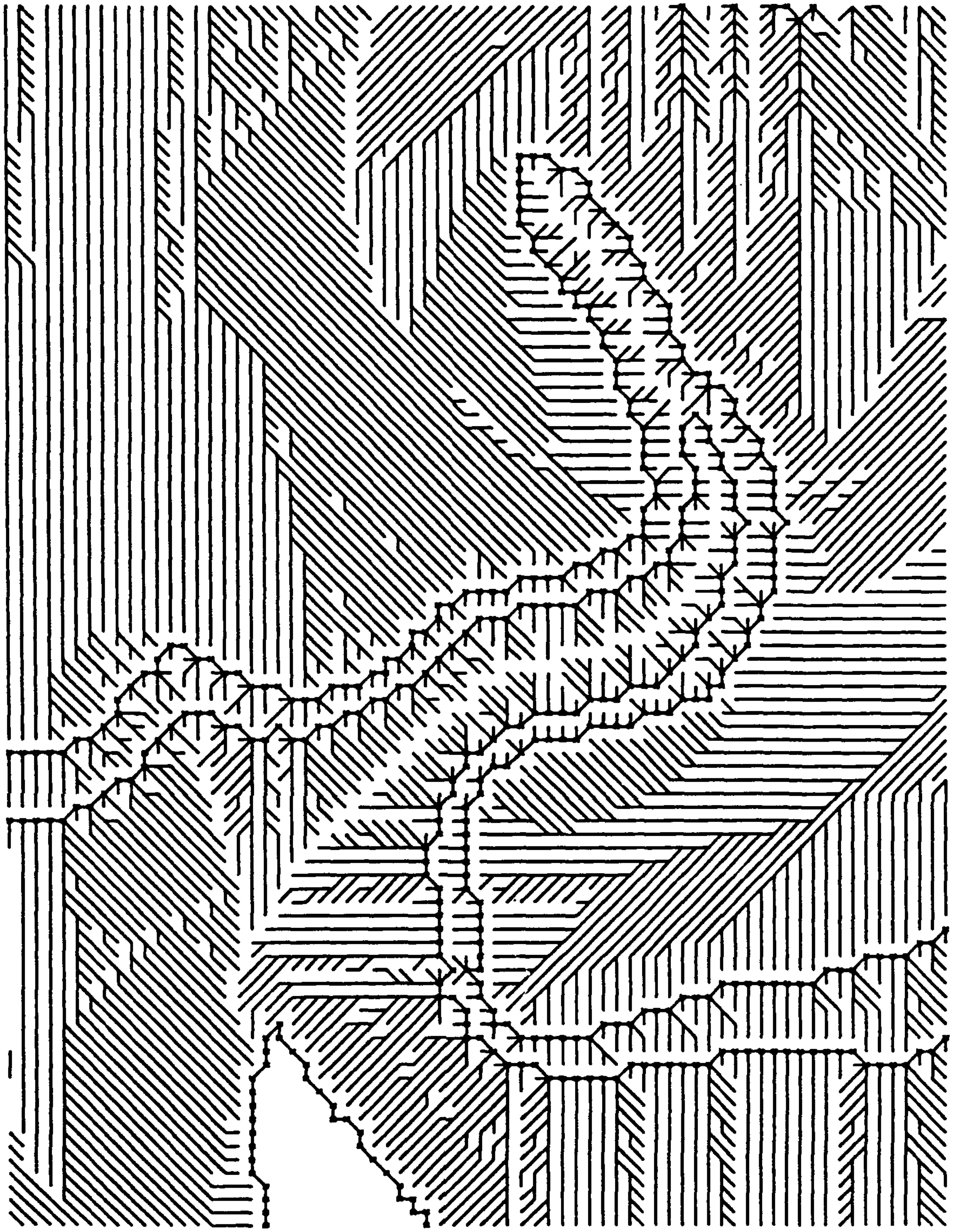

Figure 26: Enlarged portion of Figure 25. Note that the previous valley now appears as a ridge.

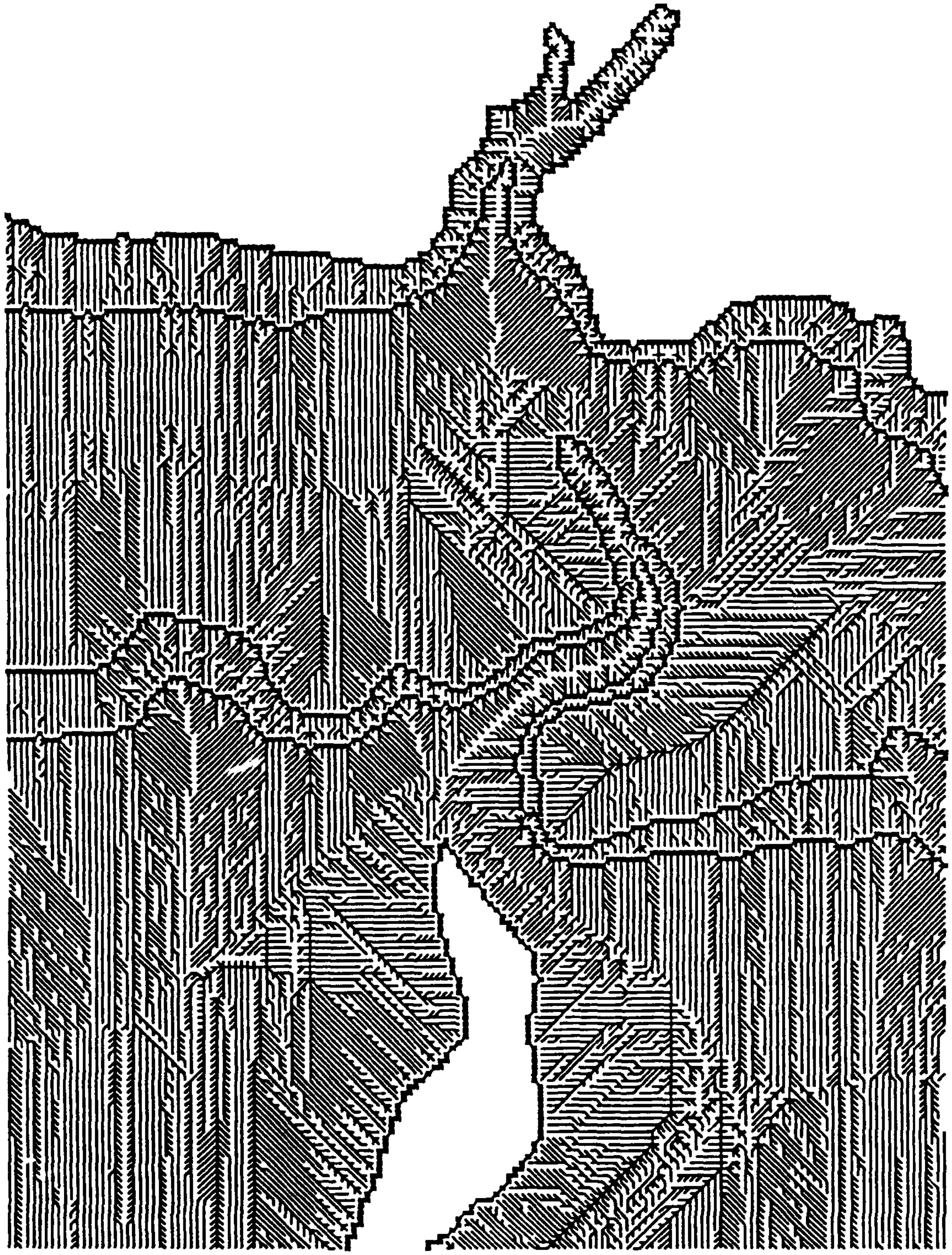

Figure 27: Fifth and final drain pattern generated from the indicated c ntours (see Figure 24) under pattern reversal. A new elevation matrix has been created by averaging the elevations derived by prorating over the five drain pattern of the iteration.

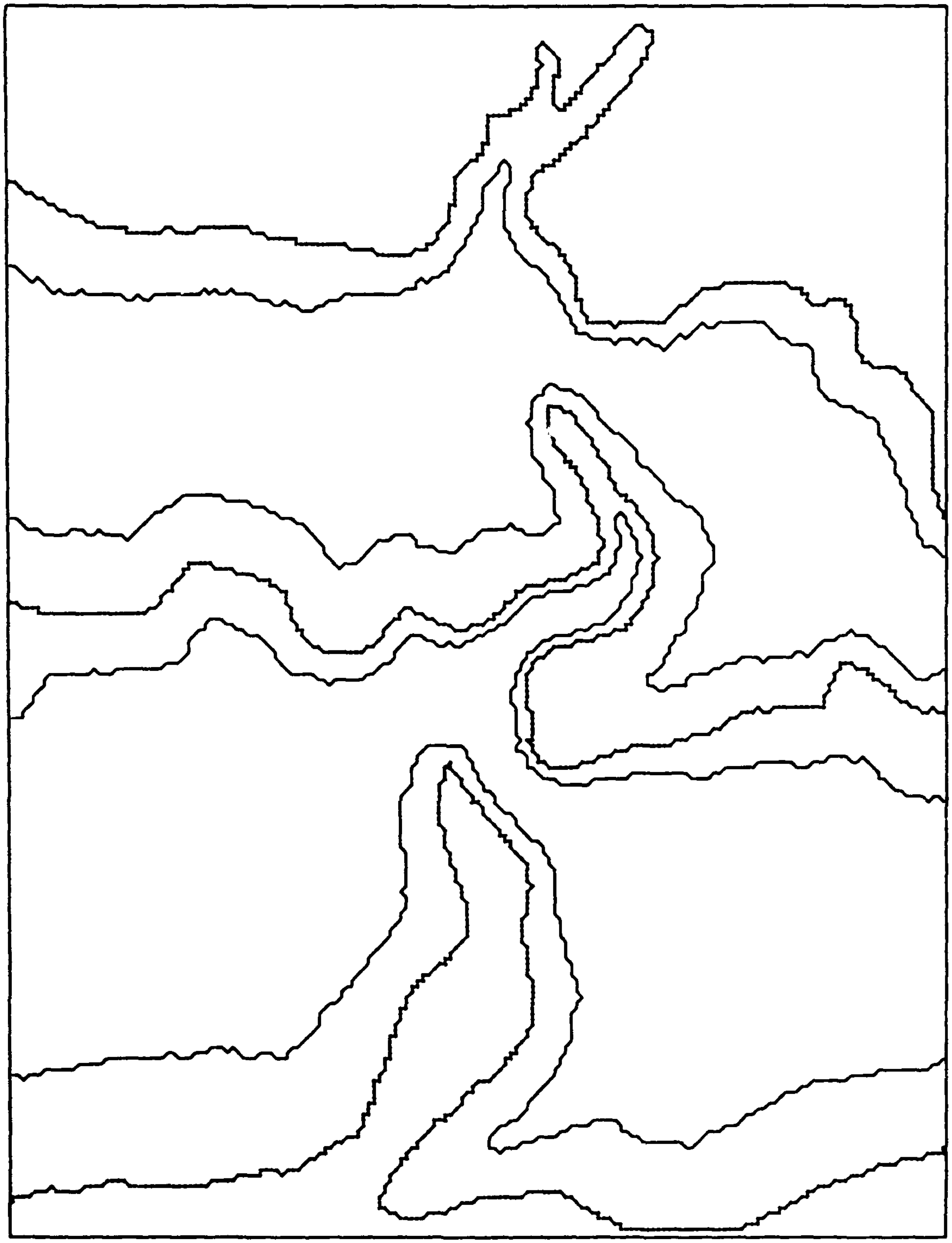

Figure 28: Two new contour lines at 4505ft and 4530ft have been extracted from the latest elevation matrix and interspersed between the previous ones. The third step of elevation generation will be based on these contours.

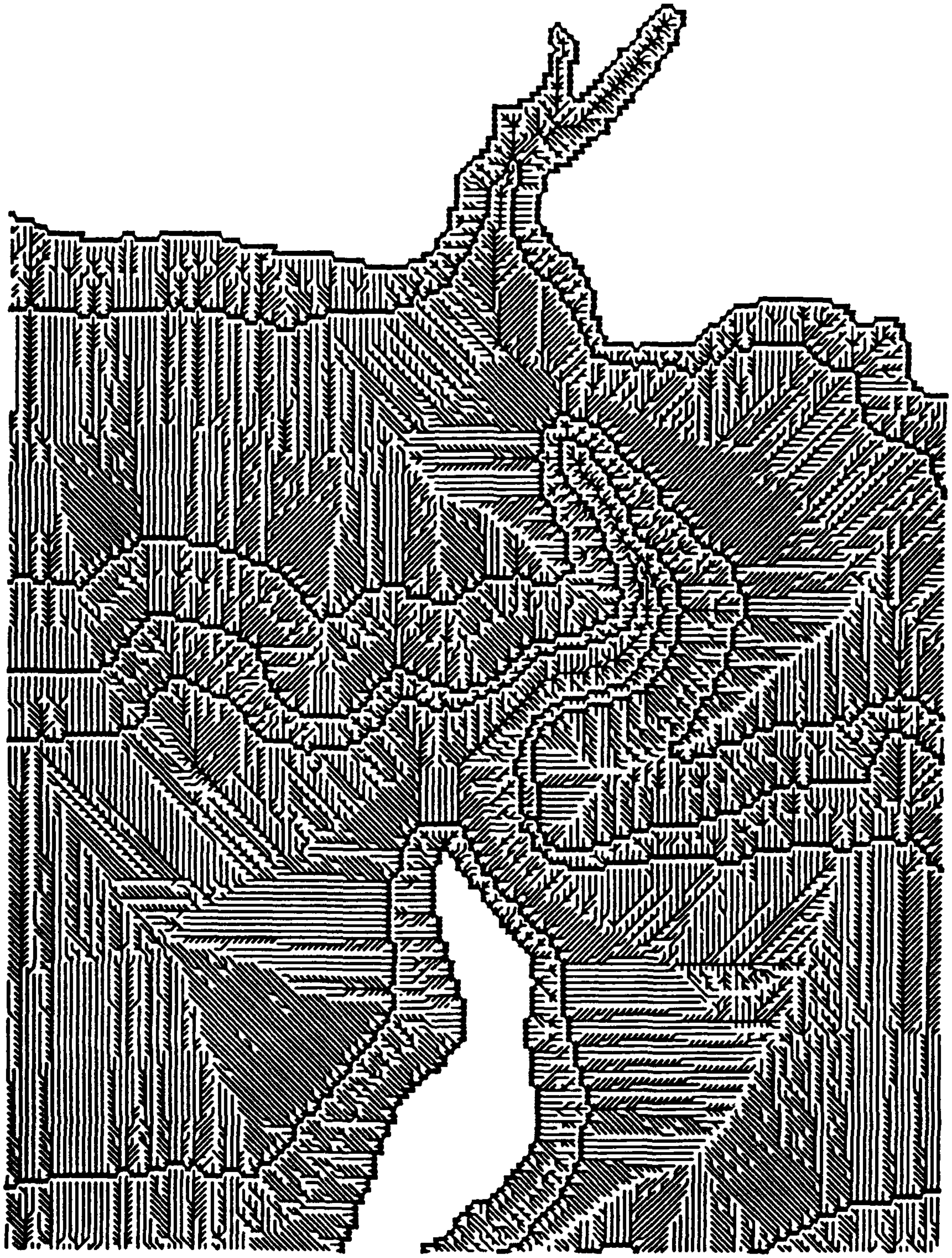

Figure 29: Five pattern iterations - without pattern reversal - yield this drain pattern based on the indicated contour lines (see also Figure 28). An elevation matrix has been again created by averaging prorated elevations. This completes the third step of elevation generation.

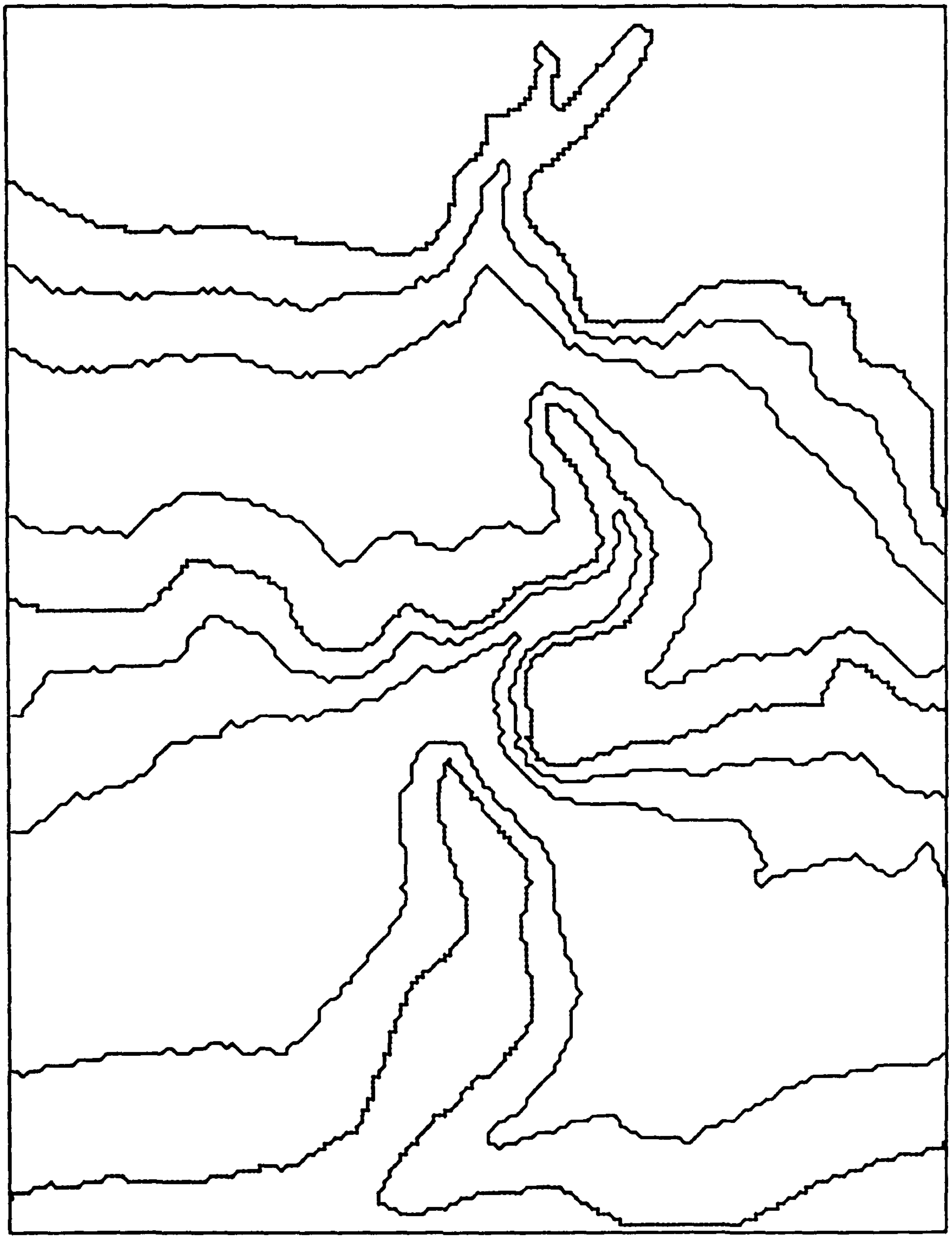

Figure 30: Additional contours at 4515ft and 4540ft have been extracted from the latest elevation matrix and interspersed between the previous ones. This initiates the fourth step of elevation generation.

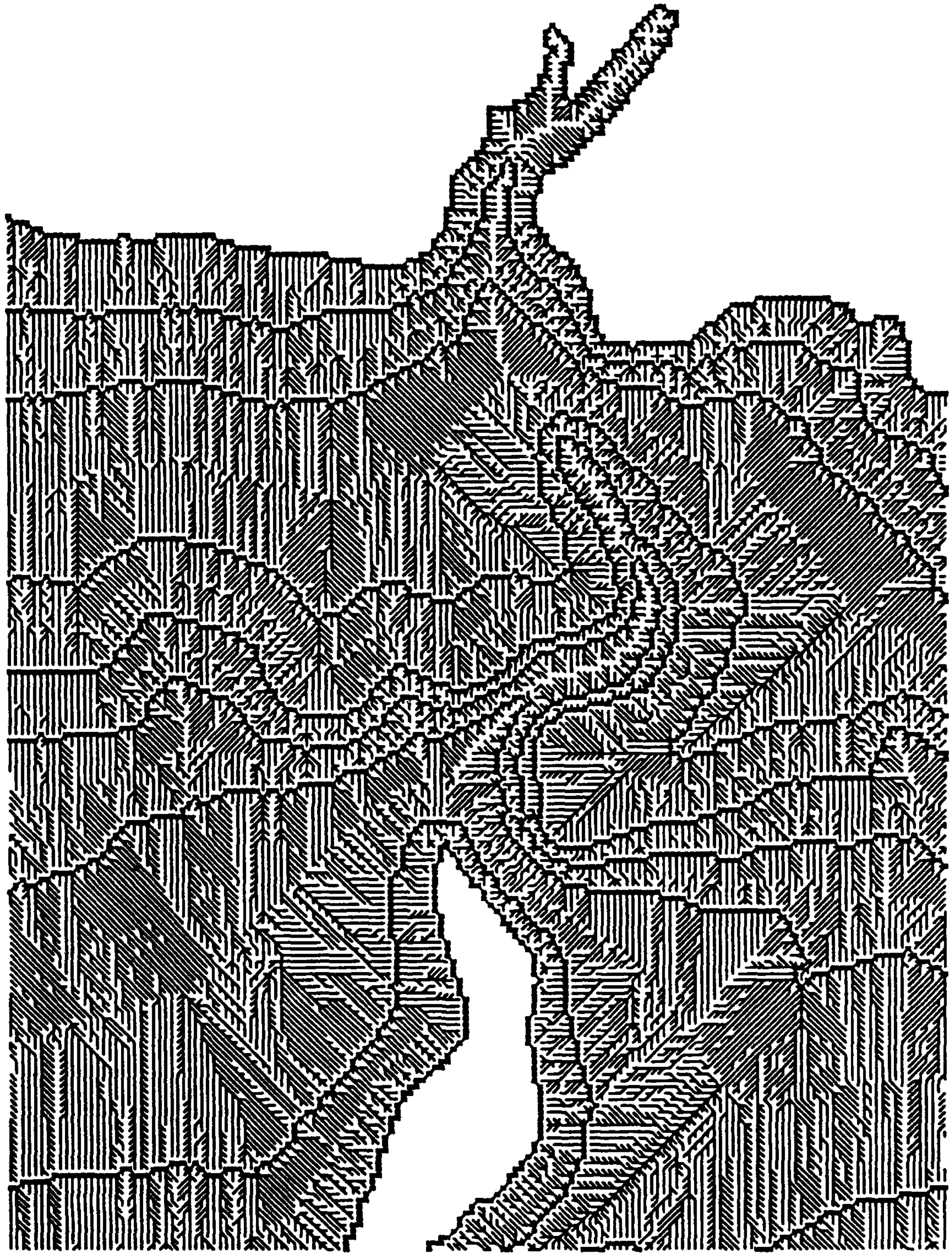

Figure 31: The fifth pattern iteration under pattern reversal based on the indicated contour lines (see also Figure 30) generates the pattern shown here and completes the fourth step of elevation generation.

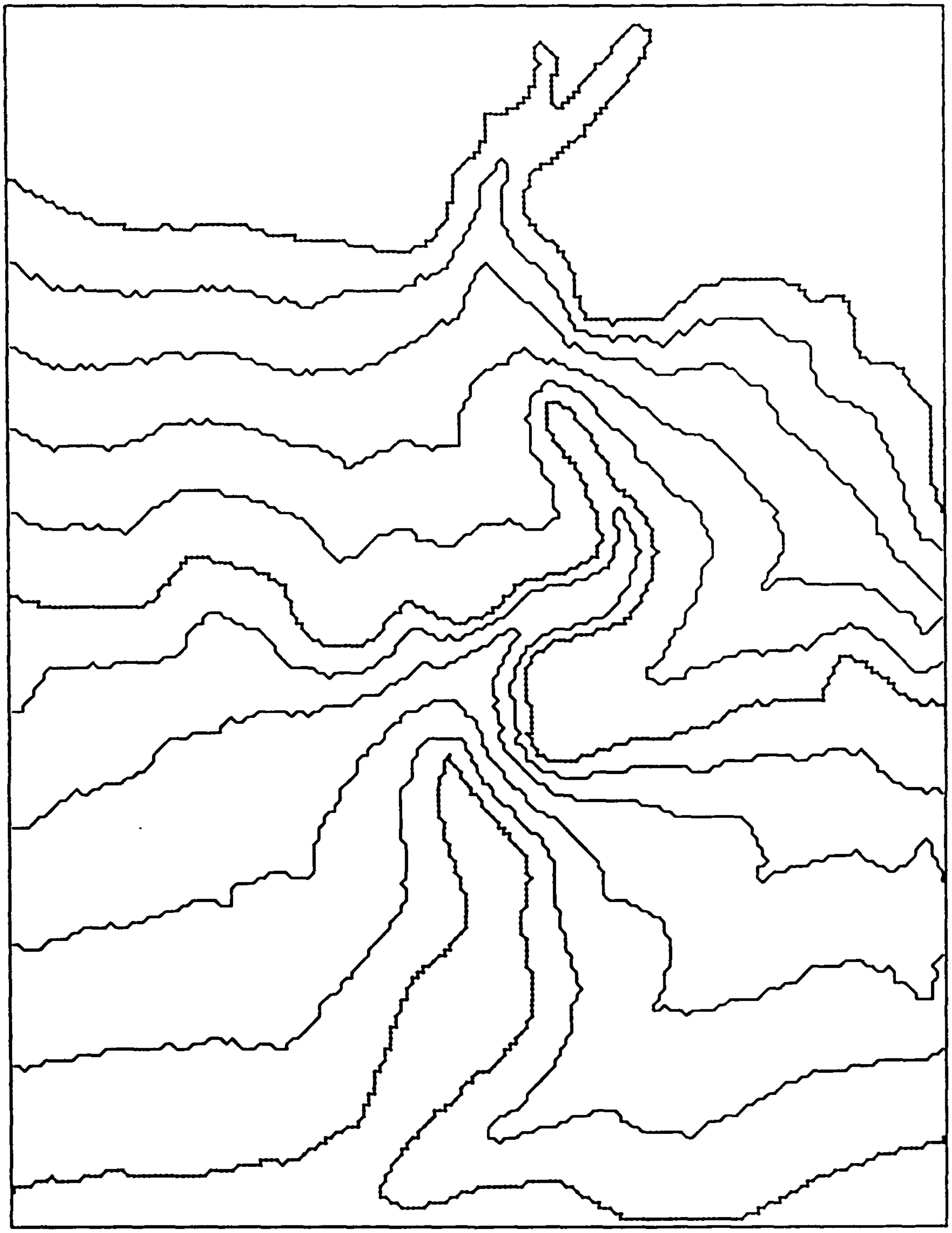

Figure 32: Having extracted contours at 4510ft and 4535ft, a full complement of contours at elevation intervals of five feet has been determined. The final step of the elevation generation to be demonstrated will be based on these contours.

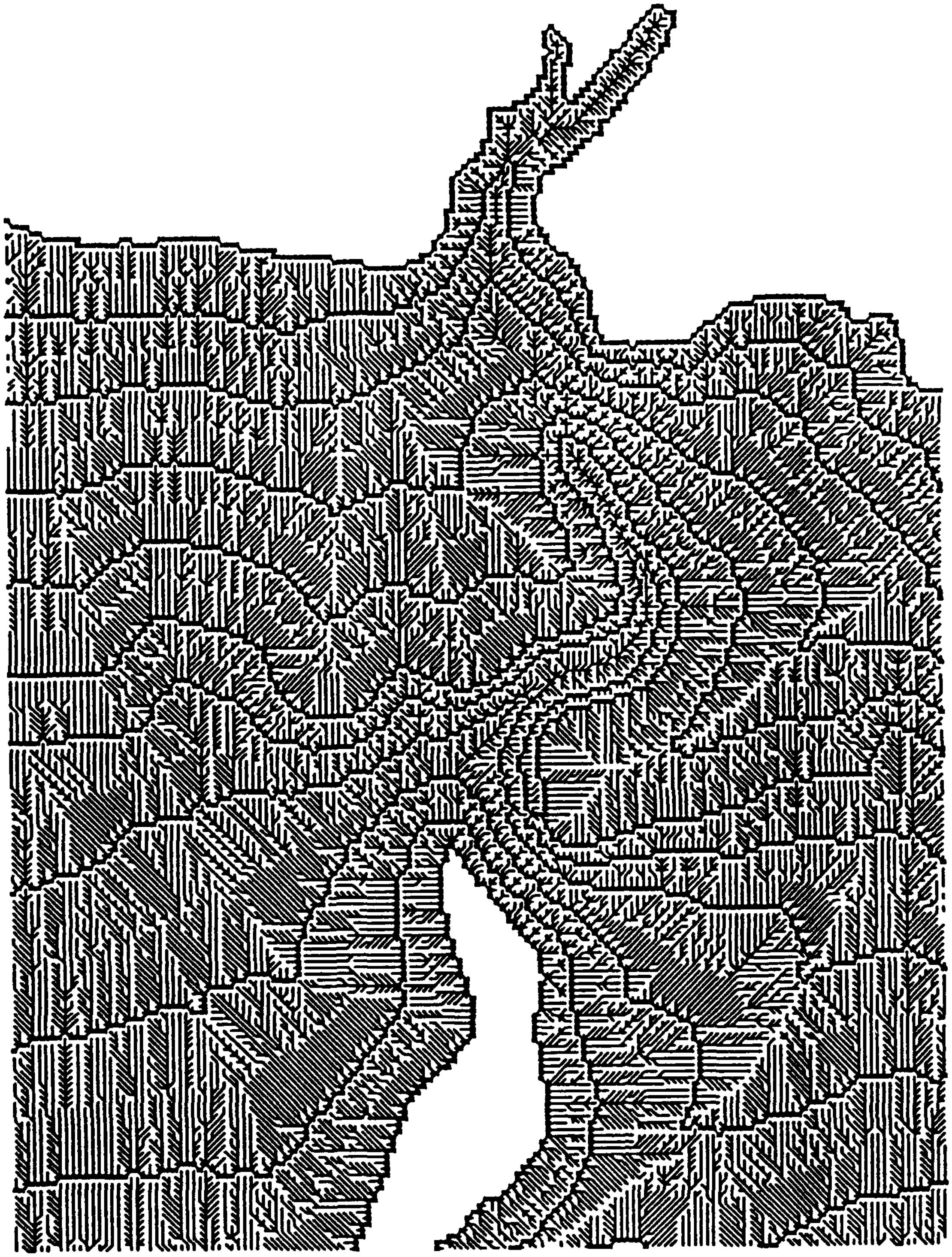

Figure 33: This drain pattern was generated by five pattern iterations based on the full complement of contour lines. As an artifact of the lower rim stabilization, the medial axis between contours is noticeable.

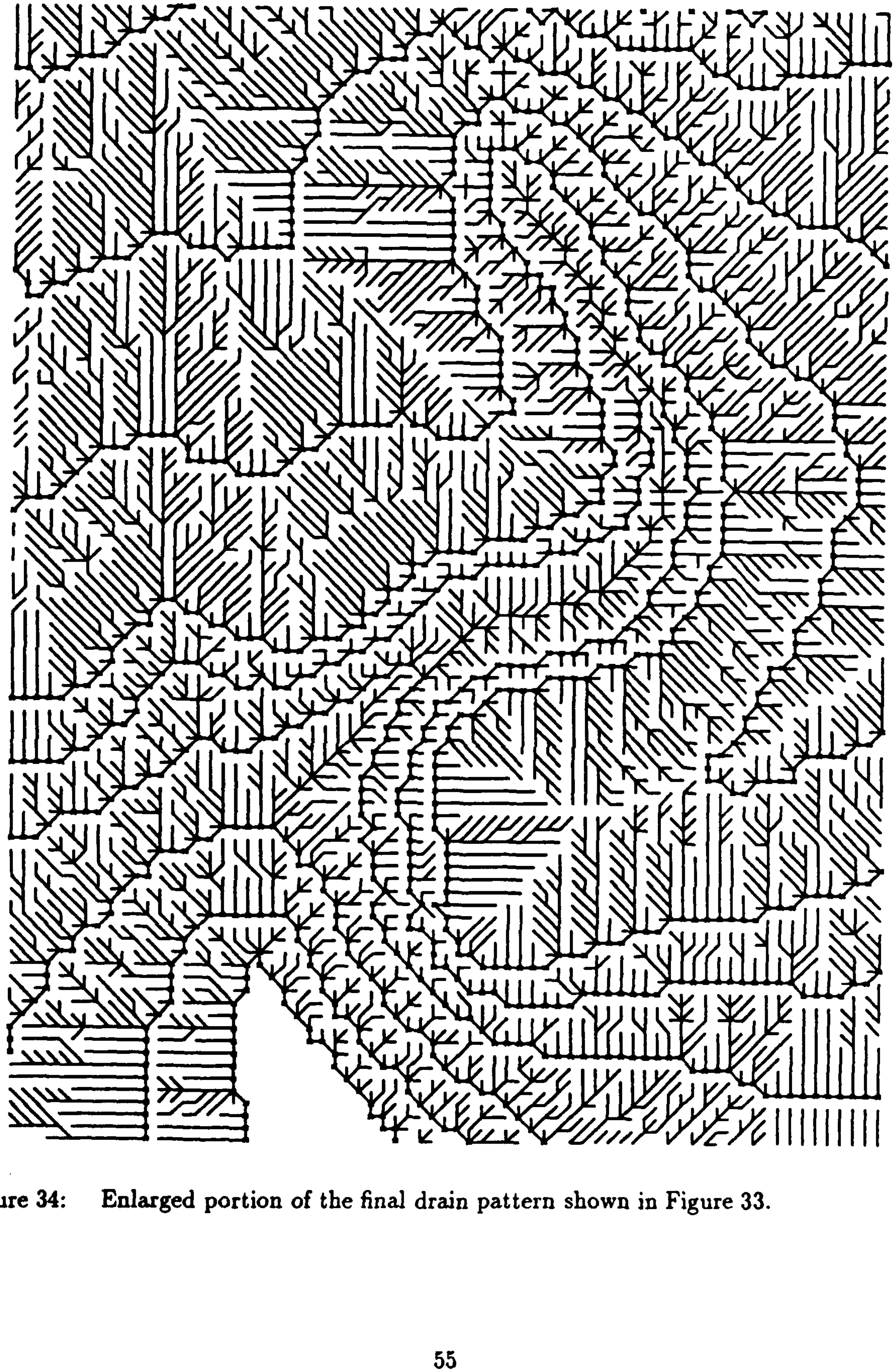

Figure 34: Enlarged portion of the final drain pattern shown in Figure 33.

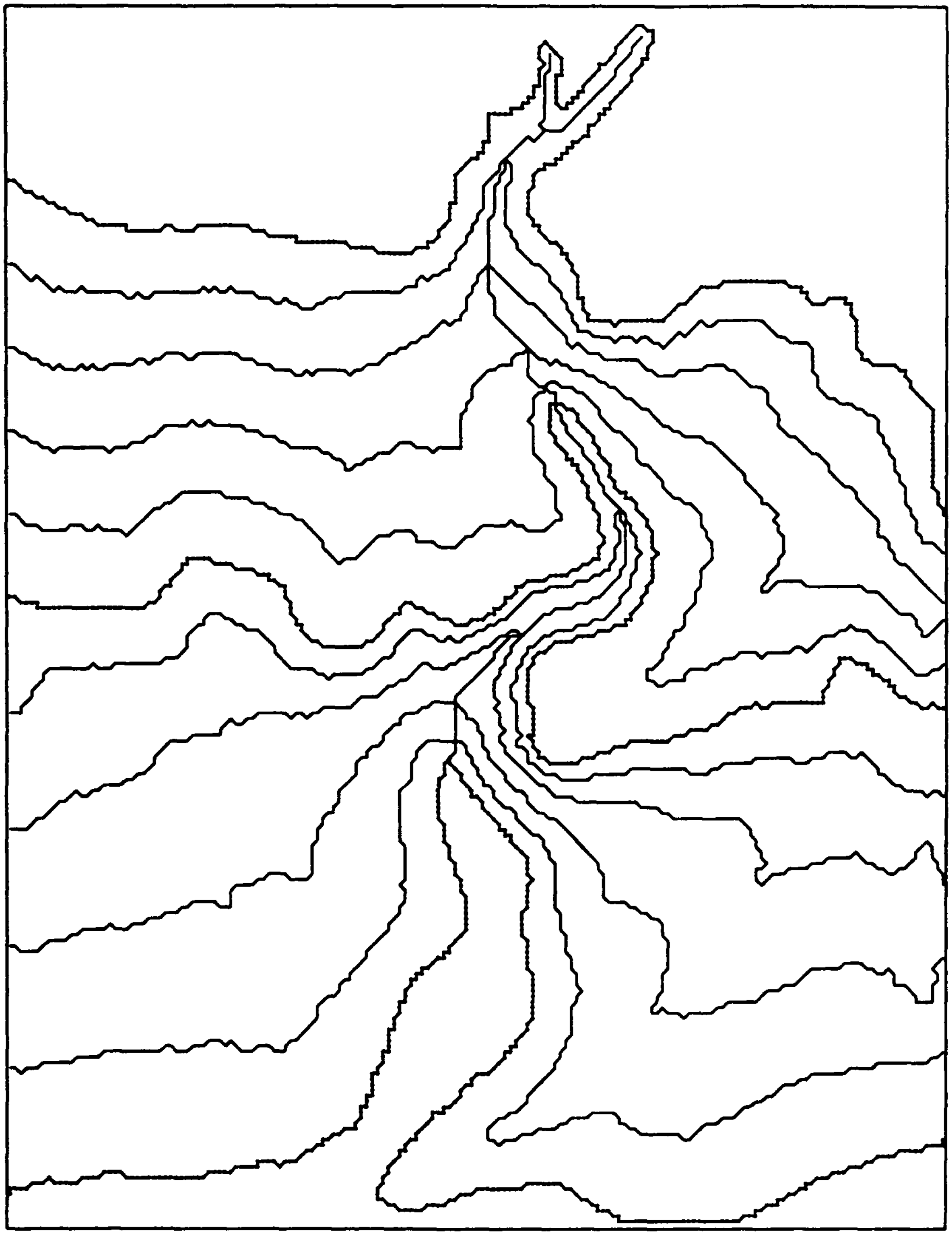

Figure 35: The major drainage line extracted from the final drainage pattern shown in Figure 33 together with the final contours shown in Figure 32. Comparison with Figure 19 reveals only minor changes in the course of the major drain.

Appendix: *PROGRAM DRAINPATH*

The calculations for the demonstration were carried out on a VAX 780 at TEC, Fort Belvoir, Virginia. The displays and illustrations were produced using the ARC/INFO system (see ESRI 1988). The software for all calculations was written in FORTRAN 77. That software is contained in two files,

DRAINPATH.FOR, WRTOUT.FOR.

File DRAINPATH.FOR contains the main

PROGRAM DRAINPATH,

which performs all calculations, and two small utility subroutines:

INTEGER*4 FUNCTION KPREDE(–, –, –), REAL*4 FUNCTION DLEDGE(–).

The first of these determines the successor in a forest pattern. The second determines the original (unadjusted) distance between two grid neighbors.

The file WRTOUT.FOR contains the

SUBROUTINE WRTPAT(–, –, –, –, –, –, –, –, –, –, –).

This subroutine handles all output functions, as well as providing the option for terminating the computation. This option is the only way in which termination is achieved.

In addition, file CONTOURING.FOR contains

PROGRAM CONTOURING,

the software used for extracting contour lines at specified elevations from elevation matrices. Two additional files, SPURCONT.FOR and ADDCONT.FOR feature utility programs for editing contours and adding them to the contour data base, respectively.

In what follows, we will outline the main program, and provide instructions on how to use it, including a description of the output options. It should be kept in mind, that the software is experimental and not intended for a production environment.

A.1 Input. The program expects input in two files,

CONT.DAT, RAND.DAT.

Further input is prompted. The file CONT.DAT contains all contour points. The File RAND.DAT identifies endpoints of contours, and the sides of the contours from which the adjusted shortest paths for drain pattern generation originate. Specifying the sides determines the regions in which contours act as source contours or root contours, respectively. This specification is therefore a vehicle for pattern reversal as described in Section 3.

In order to describe the input format more precisely, we list three different ways to identify grid points. First we have the "natural" coordinates (j,i) which start with values 1 at the lower left (southwest) corner, and which assume length 1 for the grid unit. Thus i denotes the row number and j the column number of the grid point. Second we have the "running number" k which starts with 1 at the southwest corner and continues by rows. If $jcol$ denotes the number of columns, and $irow$ the number of rows in the grid, then $k = j + (i-1) * jcol$ is the running number of the grid point (j,i). Finally, we have the "actual" coordinates inherited from a larger original grid. These coordinates are also in grid units of 1. However, they are counted from the northwest rather than the southwest corner, and the coordinates $(jeas, inor)$ of that point are input from the terminal following a prompt by the program.

The first line of the file CONT.DAT contains three integers, $nkos$=number of entries (contour points) to follow after the first line, $irow$=number of grid rows, $jcol$=number of grid columns. The first line is followed by a list of consecutive integers, ten to a line (FORMAT(10(1X,I7))). These integers are running numbers of contour points. There is no separator between the contour lines: the last point of one contour is followed by the first point of the next contour. The sequence in which the contour lines are input, and the order in which they are traversed are both immaterial.

The first line of the file RAND.DAT contains the number of boundary points of contour lines given in file CONT.DAT. Since the program is not yet set up to process closed contours, the number of boundary points is twice the number of contours. For each boundary point, a single line of information is provided. This information consists first of the actual grid coordinates $jboun(n), ibounn(n)$ of boundary point n. Each boundary point has two boundary neighbors. We are interested in that boundary neighbor $\bar{k}$ which lies on the "active" side of the contour line to which the boundary point belongs, that is, on the side into which the drain pattern rooted at the contour line spreads. We call this neighbor $\bar{k}$ the "active neighbor" of boundary point k. The second specification then gives the compass direction $charn(n) = N, W, S, E$ from the active neighbor $\bar{k}$ to the boundary point k. This information provides the orientation of the drain pattern (= normal vs. reversed). Next it is recommended to specify the integer 1 in order to indicate that the contour point next to the specified boundary point does not belong to the boundary, and to verify that the contours in CONT.DAT have that property. We also indicate the direction, clockwise or counterclockwise, in which the boundary point k is followed by its active neighbor $\bar{k}$: $clockn(n) = .true.$ indicates "clockwise". Finally, we specify the elevation $elevn(n)$ of the boundary point n. All six quantities are given in one line (FORMAT(1X,I7,1X,I7,6X,A2,I8,7X,L1,F8.1)). The sequence, in which the boundary points are represented is immaterial.

A.2 Procedure. Program DRAINPATH is outlined on a step by step basis. Referring to a current listing (Witzgall and Karalus 1991), the program portion corresponding to each step is identified by line identifiers approximately marking the beginning and the end of that portion. By and large, the following naming convention is followed: variables starting with the letter k signify running numbers of grid points, j horizontal grid coordinates, i vertical grid coordinates, d distances, h dead ends, l heap entries (see step 2 below), and ch two characters for the eight compass directions: E, NE, N, NW, W, SW, S, SE. Termination is only through exercising the termination option of the output program, which is entered at regular intervals.

1. *Input of contour information:.* The first prompt is for the actual coordinates of the northwest corner. The user is then given an option to print boundary point information in the diagnostic file DIAG.DAT. Both input files are then read and processed. At the end of this step, the output program is entered, providing a option to print out contour lines in ARCINFO format. (start-1011)

2. *Distances from contours:* The – unadjusted – grid distances from nearest contours are determined using Dijkstra's method with heap sort (see for instance Aho, Hopcroft, and Ullman 1976). The major variables are: $distk(k)$ = distance of grid point k from contour; $chark(k)$ = compass direction of successor in shortest path; $lseqk(k)$ = heap position of grid point k; $kseql(l)$ = grid point in heap position l. Contour points are characterized by $chark(k) = --$. The output program is called at the end of this step, providing the option to print the distances and the underlying shortest path pattern. (1105-3000)

3. *Initiate drain pattern along sides of contours:* The drain pattern is initiated first for all immediate neighbors which have distance 1 from the desired side of a contour. In a second pass through all contour points, those immediate grid neighbors which have distance $\sqrt{2}$ from their closest contour are included in the pattern. These separate initiation steps serve the purpose of making sure that the drain pattern, as it grows, does not spread to the wrong side of a contour. (4105-4500)

4. *Determine initial drain pattern:* The initial drain pattern is the shortest path forest with respect to the penalty-adjusted grid metric rooted on the specified side of contours. Starting with the rudimentary pattern specified in the previous step, Dijkstra's method with heap sort is again employed. Major variables are $dpenk(k)$ = penalty-adjusted distance of grid point k; $chark(k)$ = compass direction of successor in pattern; $lseqk(k)$ = heap position of grid point k; $kseql(l)$ = grid point in heap position l; $chark(k) = --$ again characterizes contour points. $chark(k)$ is blank for grid points not included into the pattern. $lseqk(k) = -1$ precludes further processing of grid point k. $lseqk(k) = 0$ holds for all grid points that have yet to enter the heap. (5000-5310)

5. *Initialize pattern iteration counter:* $irou = 0$.

6. *Increment pattern iteration counter:* $irou = irou + 1$. (5350)

7. *Weights for averaging:* The weights $roul$ and $roua$ are determined from the pattern iteration counter. The initial setting is $roul = 1.0$ and $roua = 0.0$. $roul$ weighs the prorated elevations, $roua$ the average of previously prorated elevations. (5350-5400)

8. *Calculate actual distances to contours:* Preparatory to elevation prorating by distances, the actual distances along drain paths to their root are calculated for each grid point k in the drain pattern. These distances are recorded in $dpthk(k)$. The process uses a stack $kstak(k)$ with starter $kstk$ to store drain pattern nodes whose distance has been determined recently. This stack initially contains all contour points. For each grid point removed from the stack, the distance of its predecessor nodes is determined, and these grid points are in turn placed on the stack. The output program is entered, permitting output of the drain pattern in various formats and of its actual distances. (5400-5500)

9. *Stack dead ends:* All dead ends of the drain pattern are placed on the stack $kstak(k)$ with starter $kstk$. (5540-5600)

10. *Determine lower rim of drain pattern:* Node k of the drain pattern is in the lower rim if $distk(k) = dpthk(k)$. We indicate it by setting $lrimk(k) = .true.$ (5620-5680)

11. *Initiate elevation prorating:* The two elevation arrays $elevk(k)$ and $eavek(k)$, for prorated and averaged elevations respectively, are set for contour points at their contour elevations. (6000-6100)

12. *Initialize elevation iteration counter:* $iter = 0$

13. *Increment elevation iteration counter:* $iter = iter + 1$ (6510)

14. *Examine dead end for contact:* Each dead end in stack $kstak(k)$ is examined via the contact criterion whether its elevation can be set. Dead ends found in contact are removed from the stack and saved: $kdeah(h)$ is the dead end node; $kcnnh(h)$ is its contact point; $dcnnh(h)$ is the length of the link from $kcnnh(h)$ to its grid neighbor $kdeah(h)$. If the stack is empty or if the stack contains only dead ends unable to make contact, terminate the elevation iteration. Go to Step 17. (6600-6700)

15. *Sort dead ends:* The dead ends listed in the previous step are sorted by decreasing path distance $dcnnh(h) + dpthk(kdeah(h))$. This is the length of the path from $kcnnh(h)$ to $kdeah(h)$ and from there via the drain path to the contour. (6700-6710)

16. *prorate elevations:* The dead ends are processed in order. For each dead end h, the "prorating path", from the contact point $kcnnh(h)$ at elevation $ehea$ to the dead end node $kdeah(h)$ and from there along the drain path to the first instance of an elevation $etai$ set previously, is determined, along with the length $dtot$ of this path, taking descent restriction into account: the length of arcs, not in the lower rim and moving towards the wrong contour, is weighed at 0.01 of their actual length. In a second pass, elevations are prorated along the prorating path between elevations $ehea$ and $etai$ depending on their respective distances in the prorating path. The prorated elevations are recorded in $elevk(k)$. At the same time as they are set, the prorated elevations are averaged with the previous elevations $eavek(k)$ and the result recorded again in $eavak(k)$. After all listed dead ends have been processed, go to Step 13. (6710-6800)

17. *Discard residual dead ends:* Dead ends on the stack $kstak(k)$ for which no contact could be established after repeated elevation iterations, are discarded. This terminates the elevation iteration. The output program is entered twice, permitting output of the prorated and the averaged elevations, respectively. (6910-6925)

18. *Derive drain pattern:* A new drain pattern is derived from the averaged elevations $eavek(k)$ by identifying successor grid points of steepest positive descent. If its grid neighbors are of higher or equal elevation, then a grid point is marked as a "sink": $chark(k) = **$. (6925-7100)

19. *Remove sinks:* Sinks are removed by finding an "overflow path". (7100-7180)

20. *Save old elev[illegible]ons:* The elevations *e*[illegible](k) are saved in the array $eoldk(k)$ for subsequent comparison. (7187)

21. *Check for cross-over:* The drain pattern is checked for cross-over, and corrections are made if necessary. The construction of the new drain pattern is now complete, and a new pattern iteration commences. Go to Step 6. (7190-7200)

A.3 Output The following prompts are encountered when the output program has been entered:

1. *write-out program entered - want to exit?(y/Y)*

A positive answer causes return to the main program. Otherwise, the full grid specification in actual coordinates is displayed on the screen in order to guide the window specification solicited below.

2. *x,y-window specification: (4 integers)*

The x-coordinates of the left and right window boundaries are followed by the y-coordinates of the lower and upper window boundaries. The subsequent prompt:

3. *do u want to respecify?(y/Y)*

permits a repeat of the window specification in case of error.

4. *do u want row output?(y/Y)*

There are several output modes. The "row output" mode is designed for pattern information. The pattern is characterized by assigning to each grid point a compass direction:

$$E, NE,\ N, NW,\ W, SW,\ S, SE$$

that points to the pattern successor. In addition, a selected "grid variable" has been displayed by the main program prior to entering the output routine. Grid variables may be shortest distances to contours $distk(k)$ and their path pattern, actual distances in the current drain pattern $dpthk(k)$, elevations $elevk(k)$. In the row output mode, a line with the word "row" followed by the actual y-coordinate is printed for each row in the specified window [FORMAT(' row', i5)] to be followed, for each grid point of the window row, by the actual x-coordinate, the compass direction of the pattern successor, and the value of the announced grid variable [FORMAT(5(' ', i4, ' ', a2, f8.3))]. A negative response to Prompt 4 causes the program to skip ahead to Prompt 7. Otherwise,

5. *name row output file (40 characters)*

is the final prompt for row output.

6. *do u want line output?(y/Y)*

The "line output" mode produces information in terms of ARC/INFO line files. Forest (tree) patterns, cont s and individual drain lines may be recorded in this output mode. A negative response to Prompt 6 causes the program to skip ahead to Prompt 20. Otherwise the following prompts will be encountered.

7. *name line output file (40 characters)*

Having specified a row output file previously does not preclude defining a separate line output file.

8. *display turned sideways?(y/Y)*

Option to turn display by 90 degrees.

9. *do u want to skip frame?(y/Y)*

A frame would surround the grid at .5 grid units away from boundary.

10. *do u want to skip tree-data?(y/Y)*

A positive response causes the program to skip ahead to Prompt 12.

11. *do u want a quantity zero erase?(y/Y)*

Ignore this prompt. It is no longer relevant and will be removed.

12. *do u want to skip all contour data?(y/Y)*

A positive response causes the program to skip ahead to Prompt 16.

13. *do u want to skip contour crosses?(y/Y)*

A positive response causes the program to skip ahead to Prompt 15.

14. *do u want to double cross-size?(y/Y)*

Original crosses span a rectangle .30 grid units wide, .40 grid units high.

15. *do u want to skip contour lines?(y/Y)*

At issue is the drawing of contours as lines rather than marking contour points by crosses.

16. *do u want special tree lines?(y/Y)*

The opportunity is offered to trace a pattern line to its root.

17. *do u want to specify a tree line?(y/Y)*

A negative response causes termination of the process of tracing pattern lines. The program skips ahead to Prompt 20. Otherwise:

18. *specify coordinates of start point (2 integers)*

Start coordinates are specified in actual coordinates.

19. *do u want to correct specification?(y/Y)*

If "Yes", Prompt 18 is repeated.

20. *do u want to repeat?(y/Y)*

If "Yes", Prompt 2 is repeated.

21. *do u want elevation matrix for contouring?(y/Y)*

The elevation matrix is printed out in the third output mode. The first line of output records the number of rows, the number of columns, and the exaggeration factor [FORMAT(1x,i9,i10,f10.2)]. Then a header line is created for each window row, consisting of the word "row", followed by the actual y-coordinate [FORMAT(' row',i5)]. This row header line is followed by the elevations in that row [FORMAT(10(' ',i7)]. The number of these entries is determined by the window specification. The elevations, after multiplication with the exaggeration factor, are rounded to integers.

22. *name elevation matrix file (40 characters)*

Every elevation is multiplied by an "elevation factor":

23. *specify elevation factor (1 real)*

The generation of elevation output commences.

24. *do u want to terminate the run?(y/Y)*

This provides the only possibility to terminate the run. A negative response will result in a return to the main program.

References

Aho, A.V., Hopcroft, J.E., and J.D. Ullman, 1976, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Massachusetts.

Christensen, A.H.J., 1987, "Fitting a Triangulation to Contour Lines", Proceedings of AUTO-CARTO 8, pp. 57-67.

Clough, R.W. and J.L. Tocher, 1965, "Finite Element Stiffness Matrices for Analysis of Plates in Bending", Proceedings of the Conference on Matrix Methods in Structural Mechanics, Air Force Institute of Technology, Wright-Patterson A.F.B., Ohio.

Douglas, D.H., 1987, "Experiments to Locate Ridges and Channels to Create a New Type of Digital Elevation Model", Cartographica, Vol. 23, pp. 29-61; also, The Canadian Surveyor, Vol. 41, pp. 373-406.

ESRI, 1988, ARC/INFO, The Geographic Information System Software, User Guide, Vols. 1 and 2, 2-nd printing, Environmental Systems Research Institute (ESRI), 380 New York Street, Redland, CA 92373.

Grotzinger, S., Danielson, B., Caldwell, D., and B. Mandel, 1984, The Contour-to-Grid Problem. An Evaluation of Simplicially Based Local Surface Estimators (SWIRL, CONTOCES, and TRENDVOL), Interim Paper, U.S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia.

Lawson, C.L., 1977, "Software for C^1 Surface Interpolation", in Mathematical Software III, J.R. Rice, editor, Academic Press, New York.

Mandel, B., Witzgall, C. and J. Bernal, 1987, Contour-to-Grid Interpolation with Non-linear Finite Elements: A Feasibility Study, U.S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia, Report No. ETL-0472, 69 pages.

Noma, A., 1974, "Toward Creation of a Digital Terrain Data Base", Proceedings of the ACSM-ASP Annual Convention, p.218.

Peucker (Poiker), T.K. and D.H. Douglas, 1975, "Detection of Surface-Specific Points by local Parallel Processing of Discrete Terrain Elevation Data", Computer Graphics and Image Processing. Vol. 4, pp. 375-387.

Preparata F.P., and M.I. Shamos, 2nd ed., 1988, Computational Geometry, Springer-Verlag, New York.

Rinehart, R.E., and E.J. Coleman, 1988, "Digital Elevation Models Produced from Digital Line Graphs", Proceedings of the ACSM-ASPRS Annual Convention, Vol. 2, pp. 291-299.

Scarlatos, L.L., 1990, "A Refined Triangulation Hierarchy for Multiple Levels of Terrain Detail", Proceedings of the IMAGE V Conference, pp. 115-122.

Scarlatos, L.L., 1991, "Adaptive Hierarchical Triangulation", Proceedings of AUTO-CARTO 10, Vol. 6, pp. 234-246.

Seemuller, W.W., 1989, "The Extraction of Ordered Vector Drainage Networks from Elevation Data", Computer Vision, Graphics, and Image Processing, Vol. 47, pp. 45-58.

Witzgall, C., Gilsinn, J.F., and D.R. Shier, 1981, "Shortest Paths in Networks", in <u>Case Studies in Mathematical Modeling</u>, W.F. Boyce, editor, Pitman, Boston.

Witzgall, C., and R.S. Karalus, 1991, "Program DRAINPATH", FORTRAN source code listing, October version, available upon request.

Zienkiewicz, O.C., 2nd ed., 1971, <u>The Finite Element Method in Structural and Continuum Mechanics</u>, McGraw Hill, New York.